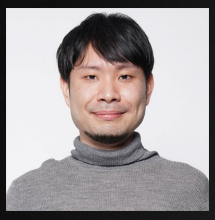
# 『グランツーリスモ７』における天球レンダリング

㈱ポリフォニー・デジタル
鈴木 健太郎
安冨 健一郎

# 自己紹介



**鈴木 健太郎**
(株)ポリフォニー・デジタル
グラフィックスチーム　リードエンジニア

**安冨 健一郎**
(株)ポリフォニー・デジタル
福岡アトリエ　グラフィックスチーム　テクニカルアーティスト

POLYPHONY DIGITAL

こんにちは

# はじめに

# グランツーリスモ7

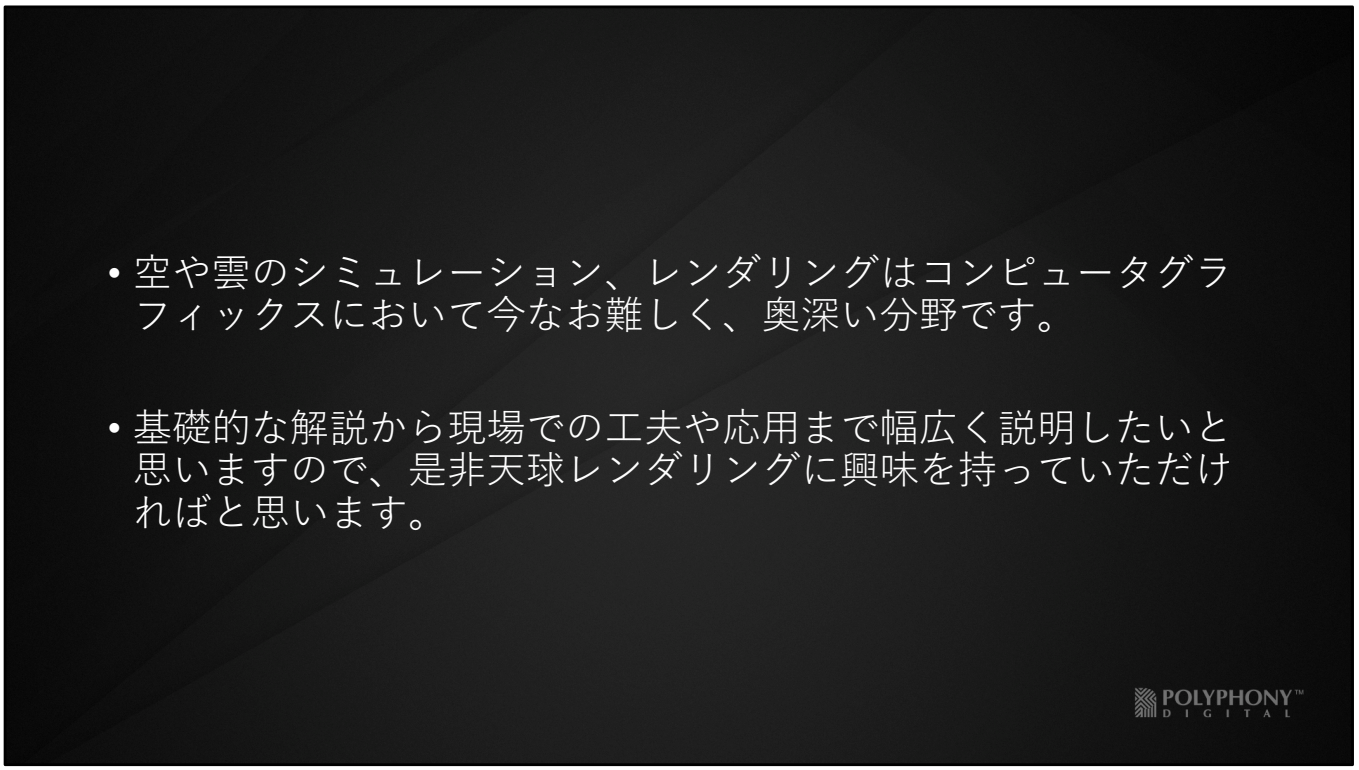
- PlayStation®4/5 用ドライビングシミュレーター
- より進化した挙動
- 400車種以上を再現
- フォトリアリスティックな映像
- 時間天候変化をサポート

POLYPHONY DIGITAL

「グランツーリスモ」は今年で２５周年を迎えたドライビングシミュレーターで、リアルな挙動と写実的なグラフィクスを特長とするレースゲームでもあります

今年３月にリリースされた最新作「グランツーリスモ7」ではリアルタイムの時間変化と天候変化が実装され、よりリアルなドライビング体験と、多様な美しい景観を実現しました。

動画です




CEDEC東京でも同じタイトルの講演をしましたが、今回は時間枠が4倍になっているので、コンテンツましましでお送りしたいと思います
すごく面白い分野なので、楽しんでもらえたらと思います

# 本日の内容

1. 空と雲の物理
2. 現代的な大気散乱シミュレーションと関連研究
3. GT7における天球レンダリング手法(オフライン)
4. GT7における天球レンダリング手法(ランタイム)
5. まとめ

POLYPHONY™ DIGITAL

本日のメニューです

まず、空と雲の基本的な物理のお話。次に、空をCGでシミュレーションする基本をお話しします
その後、グランツーリスモ7で

# 本日の内容

1. **空と雲の物理**
2. 現代的な大気散乱シミュレーションと関連研究
3. GT7における天球レンダリング手法(オフライン)
4. GT7における天球レンダリング手法(ランタイム)
5. まとめ

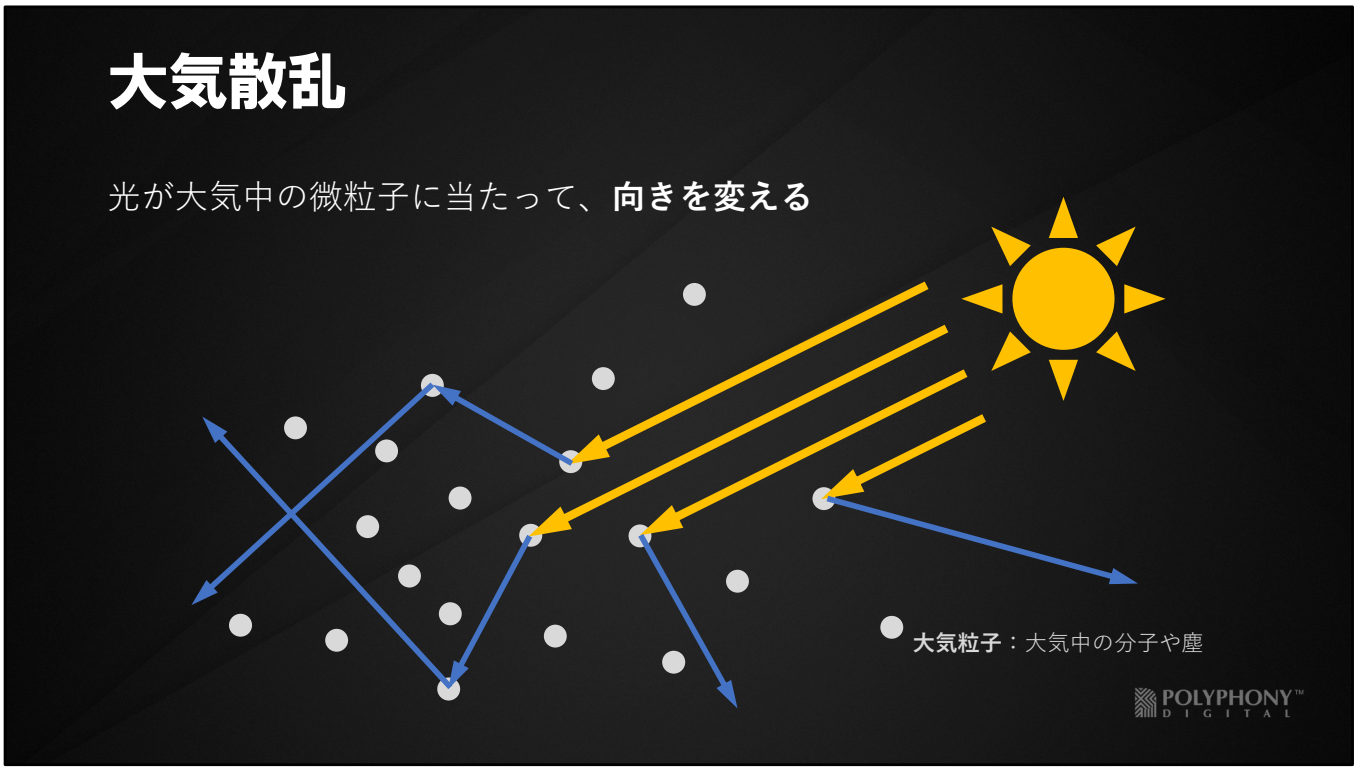
POLYPHONY™ DIGITAL

# 空と雲の物理

POLYPHONY DIGITAL

# 空の光とは何か

ふだん見慣れている空の光は、いったい何なのでしょうか？



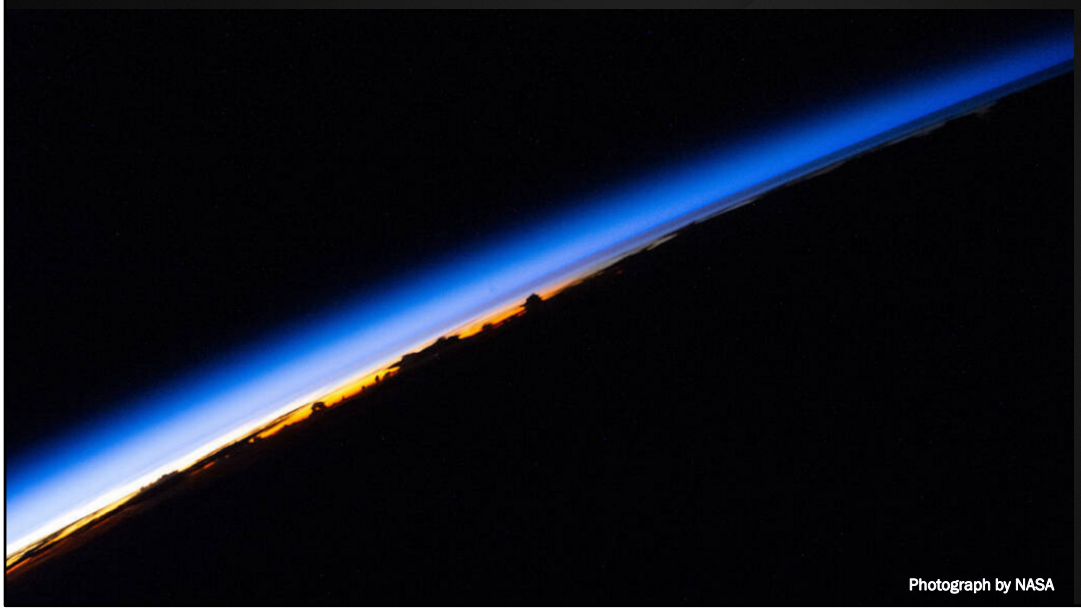

大気中には分子や塵などさまざまな微粒子が浮いています。その微粒子が太陽の光を散乱した光が空の光です

# 大気散乱

光の向きを変えるから、光が**横からも**見える

こちらに向かっていない光が見えますね
太陽光は1方向からしかやってきて直進しますが、散乱によって直射日光の方向「以外」に光がまわるようになります。この光が空を満たしています

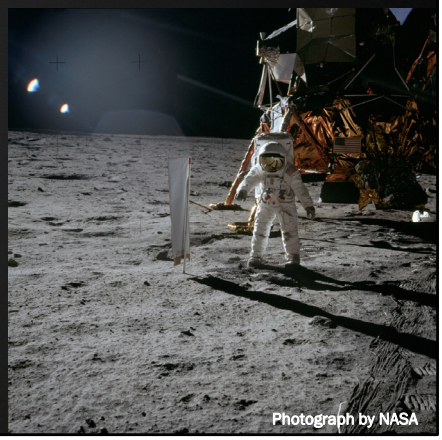
# 大気散乱



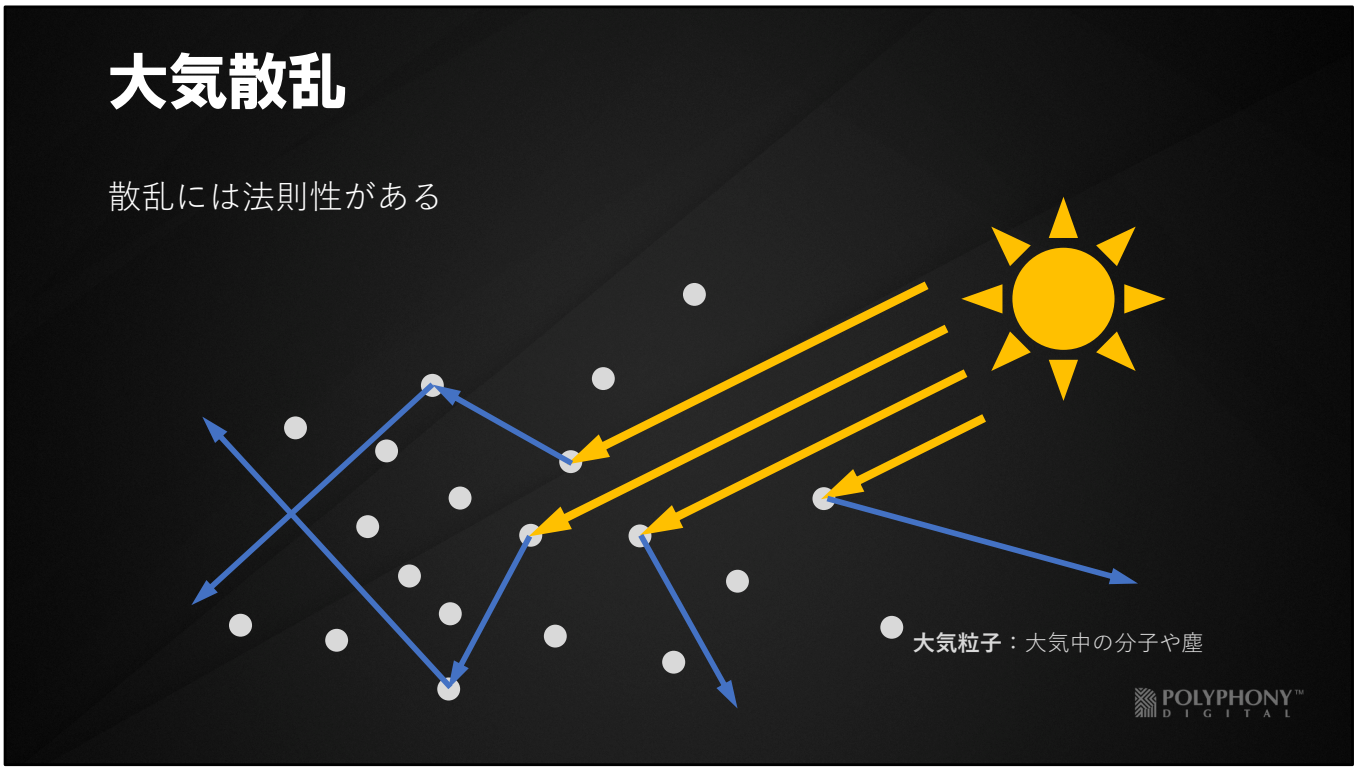

大気が太陽に照らされて、光っているわけです

# 大気のない月の空

大気の無い月は、昼間でも空は真っ暗



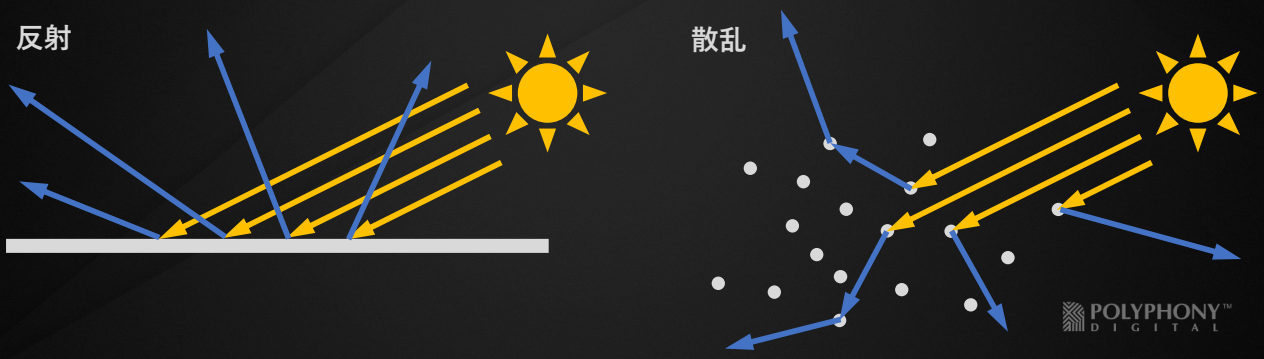

地球と違って、大気のない月には空の光がありません




では、散乱の特徴、法則についてこれからお話ししていきます

# 反射とは違うの？

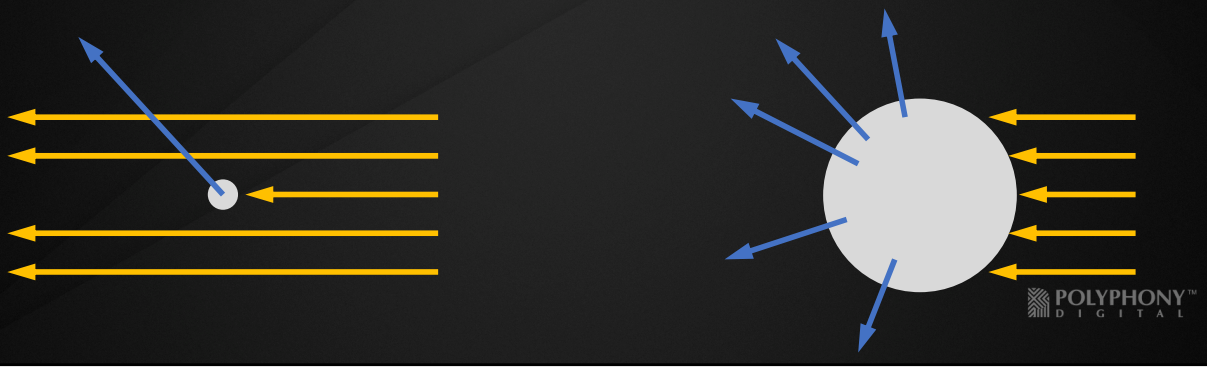
光が何かに当たって**向きが変わる**点は同じ。ただし**光の曲がり方が異なる**



# 散乱の法則（ミクロ視点）

**粒子の屈折率･大きさ**で散乱･吸収のしやすさが変わり、大きい粒子はたくさんの光を受けて、たくさん散乱･吸収する

これらの特徴は、同一粒子でも**光の波長によって異なる**




粒子によって散乱の特性は変わります

# 散乱の法則（ミクロ視点）

散乱は入射光から光を**奪って**別の方向に光を出す、という言い方もできる。
直進する入射光から**散乱・吸収**によって光が失われることを**減衰**という

粒子がなかったらそのまま直進するはずだった光は消えていることに注目してください

# 散乱の法則（ミクロ視点）

ここまでに説明したことには、以下のように名前が付いている

粒子1個あたりの

- 散乱しやすさ：**散乱断面積** (scattering cross section)
- 吸収しやすさ：**吸収断面積** (absorption cross section)

- **散乱断面積+吸収断面積 = 減衰断面積** (extinction cross section)

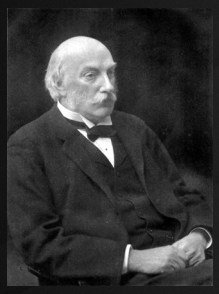
POLYPHONY DIGITAL

いちおう用語も説明しておきますが、これはメモとか取らなくてよいです
興味を持ってくれた皆さんがスライドをダウンロードして、あとから見返したときに役立つように

# 散乱の法則（ミクロ視点）

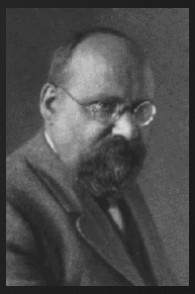
大気散乱には、大別して**レイリー散乱**と**ミー散乱**とがある。散乱は他にもいくつかあるが、大気光学現象ではこのふたつを押さえておけばOK

**レイリー散乱**
光の波長より**小さい**粒子による光散乱の法則



John William Strutt, 3rd Baron Rayleigh
(England, 1842 - 1919)

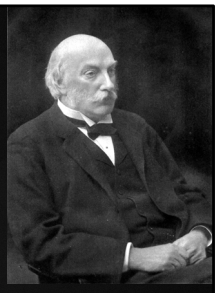
**ミー散乱**
光の波長より**大きい**粒子による光散乱の法則



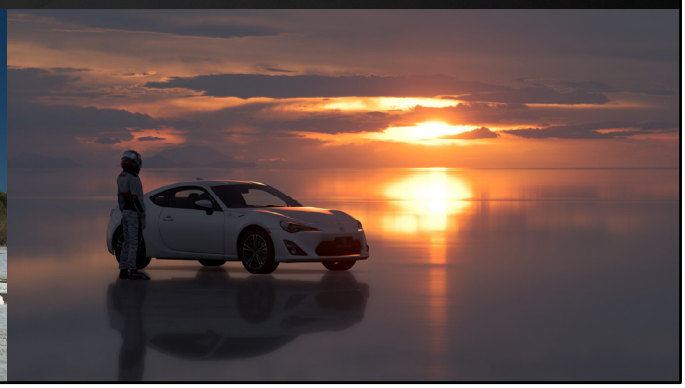
Gustav Adolf Feodor Wilhelm Ludwig Mie
(Germany, 1868 - 1957)

POLYPHONY™ DIGITAL

# レイリー散乱

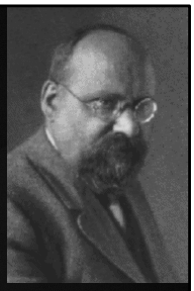


- 光の波長よりも十分に小さい粒子(約**100nm未満**)で起きる
  主に**大気分子**がこのサイズ
- 波長が短い光(青や紫)ほど強く散乱する特徴がある。**青い**。
- 空の青さ、夕陽の赤さはレイリー散乱の作用(詳しくは後述)

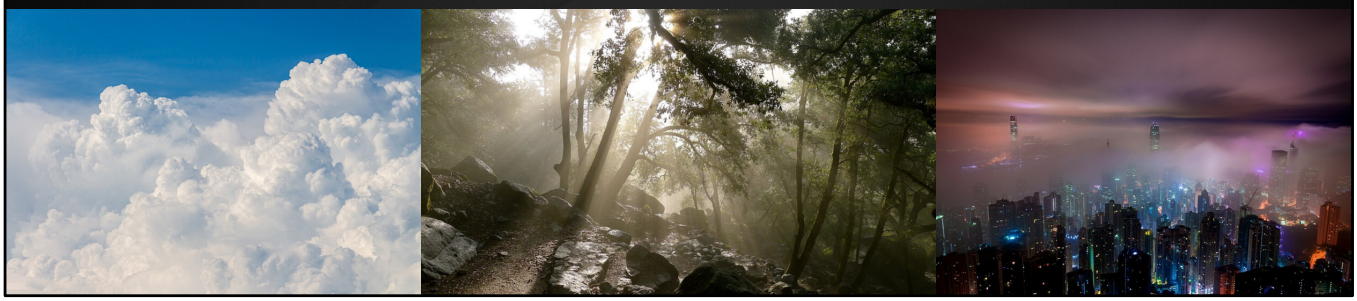


小さな粒子はレイリー散乱。青い散乱です。空の青さ、夕陽の赤さをつくっています

# ミー散乱



- 光の波長と同程度以上の大きい粒子(**約100nm-100μm** )で起きる**粉塵やスモッグ、水滴(エアロゾル)**がこのサイズ
- 波長依存性は小さく、入射光の色がほぼそのまま出る。**白い。**
- 雲や霧、チンダル現象は、主にミー散乱の作用による

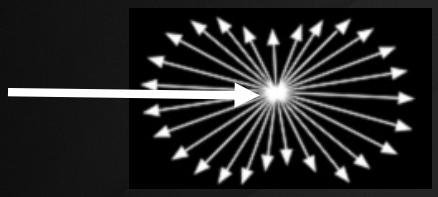


大きな粒子はミー散乱。白い散乱です

# 位相関数

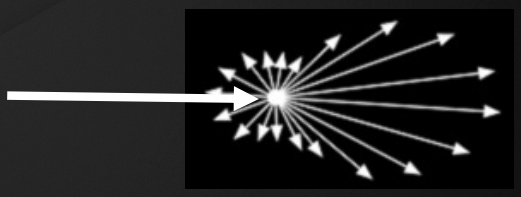
光が粒子に当たるとき、向きを変えた光がどの方向にどのくらい出るかを表す関数。これも**粒子の屈折率･大きさ**で変わり、光の波長で異なる。

**レイリー散乱**
光の波長より**小さい**粒子による光散乱。



前方・後方がやや強めで、側方は少し弱い

**ミー散乱**
光の波長より**大きい**粒子による光散乱

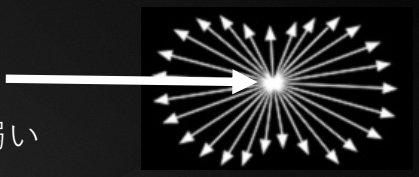


大部分が前方で、粒子が大きいほど前に偏る。
光波の干渉によって複雑な分布になる

POLYPHONY DIGITAL

光が向きを変える、とさっきからしつこく言っていますが、光がどのように向きを変えるかの法則が位相関数です。
レイリー散乱とミー散乱ではこれが違います

# 位相関数(レイリー散乱)

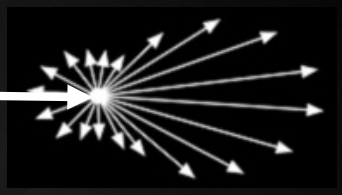


中央の暗い領域に注目。この付近の角度が散乱の弱い側方にあたる(太陽光線と直交する角度)

POLYPHONY DIGITAL

# 位相関数（ミー散乱）

逆光の雲は強い前方散乱により光が透けて、強く輝く。
それと比べると側方の雲はだいぶ薄暗い



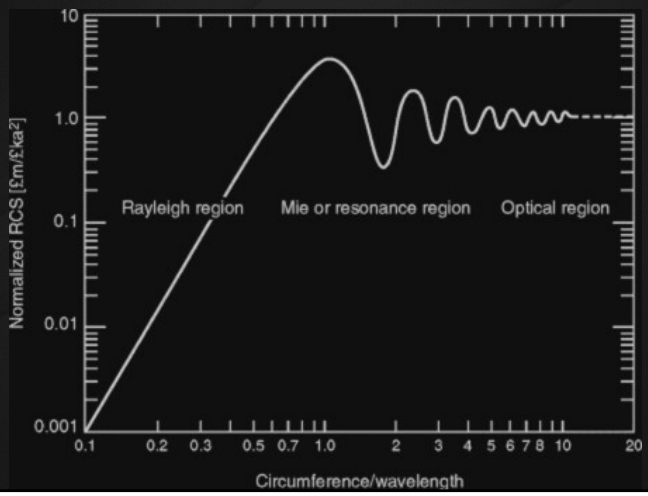
POLYPHONY DIGITAL

ミー散乱の代表は雲です。太陽の近くは光が透けて白く輝いていますが、それに比べると側方はずいぶん暗いですね

太陽を背にする向きになると、また明るくなります。

# 散乱を特徴づける要因

どんな散乱になるかは、粒子の**屈折率･大きさ･光の波長**で決まってくる



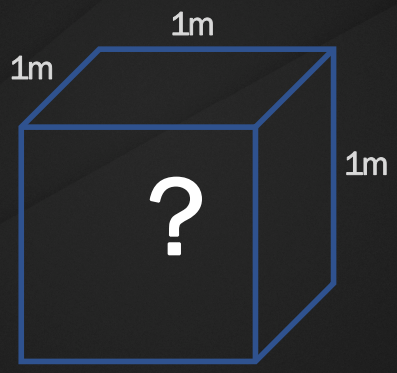

ここまででお伝えしたかったことは、粒子によって散乱の特徴が違うこと
それが粒子の屈折率と大きさ、光の波長で変わる、ということです

レイリー散乱とミー散乱はどこかを境に突然切り替わるわけではなく、この図のようになめらかにつながっています

[図:Wikipedia英語版 Mie scattering]

# 散乱の法則(マクロ視点)

ここまでは**粒子1個**のミクロなおはなし。現実の大気にはとんでもない数の多様な粒子が浮いているので、マクロな視点も重要になる



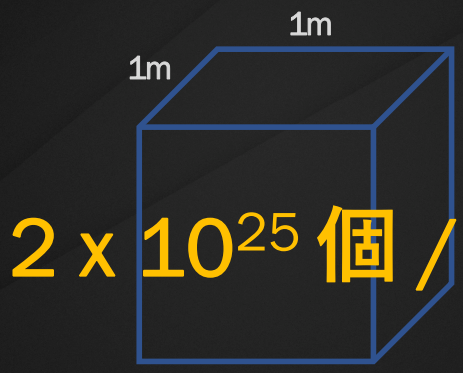

POLYPHONY DIGITAL

ここまでは粒子ひとつの話をしてきました
しかし、僕らがみている空は大量の粒子の散乱の混ざりものです

どのくらいの数かというと

# 散乱の法則（マクロ視点）

ここまでは**粒子1個**のミクロなおはなし。現実の大気にはとんでもない数の多様な粒子が浮いているので、マクロな視点も重要になる




$2 \times 10^{25}$ 個 / $m^3$

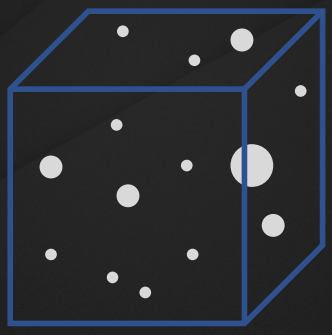
**とか、そのくらいの数**

POLYPHONY DIGITAL

このくらいです。1メートル立方の空間にこのくらい入っています。これが何百キロ広がっているので、途方もない数になります
粒子一個に注目しているだけだとキリがない

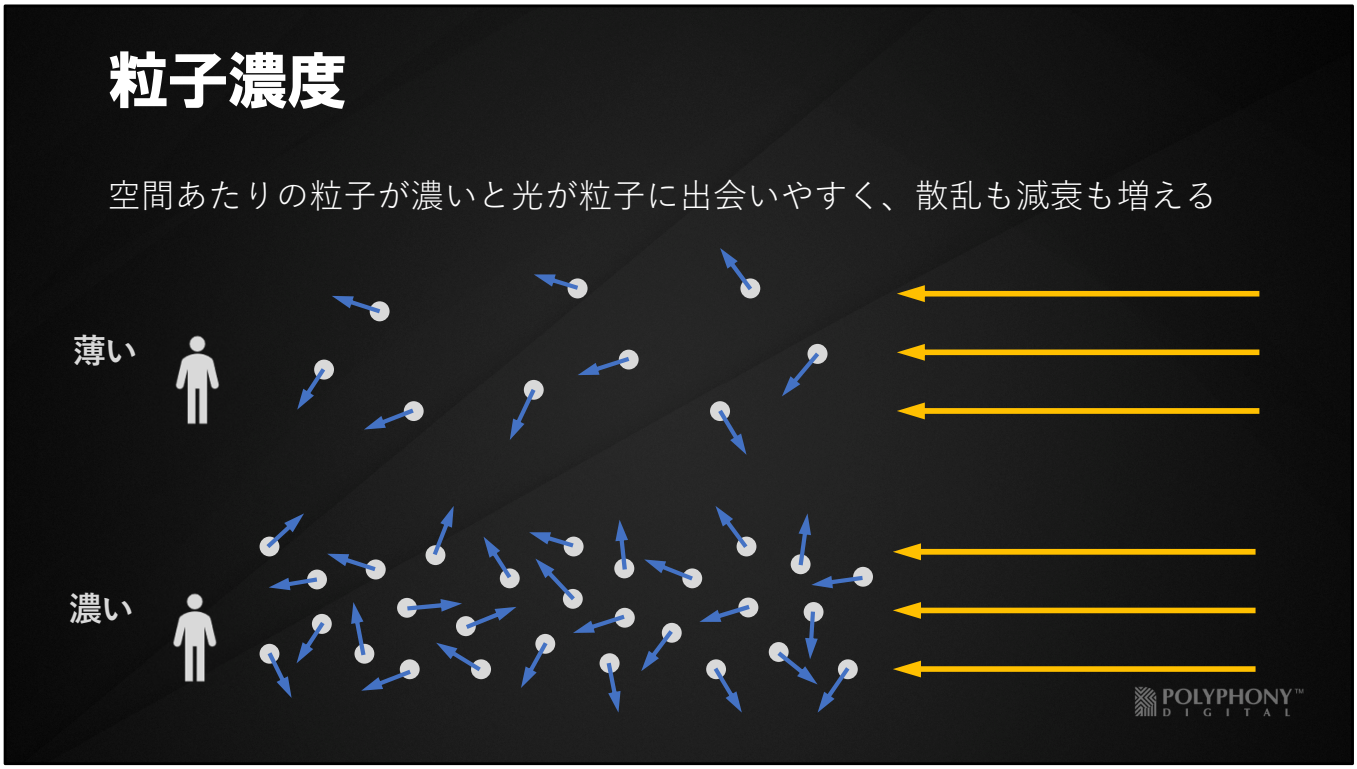
## 散乱の法則（マクロ視点）

どんな粒子が、どのくらい浮いているのか？



POLYPHONY DIGITAL

ですから、粒子じたいの特徴だけでなく、その数も考えないといけません




数の影響は直感通りです
粒子が濃いと散乱は増えます。弾幕が濃いと当たっちゃう、みたいな話です

# 粒子濃度

地球大気は**高度が高いほど薄くなる**。高い山の上や飛行機から見える空や太陽は地上とは異なる

飛行機から見える空は青黒いですよね。散乱が少ないためです

飛行機から見える太陽はやたら白くてまぶしいと思いませんか。さっき言ったことを思い出してください
散乱が少ないと、減衰も小さい。だからまぶしい

# 粒子濃度

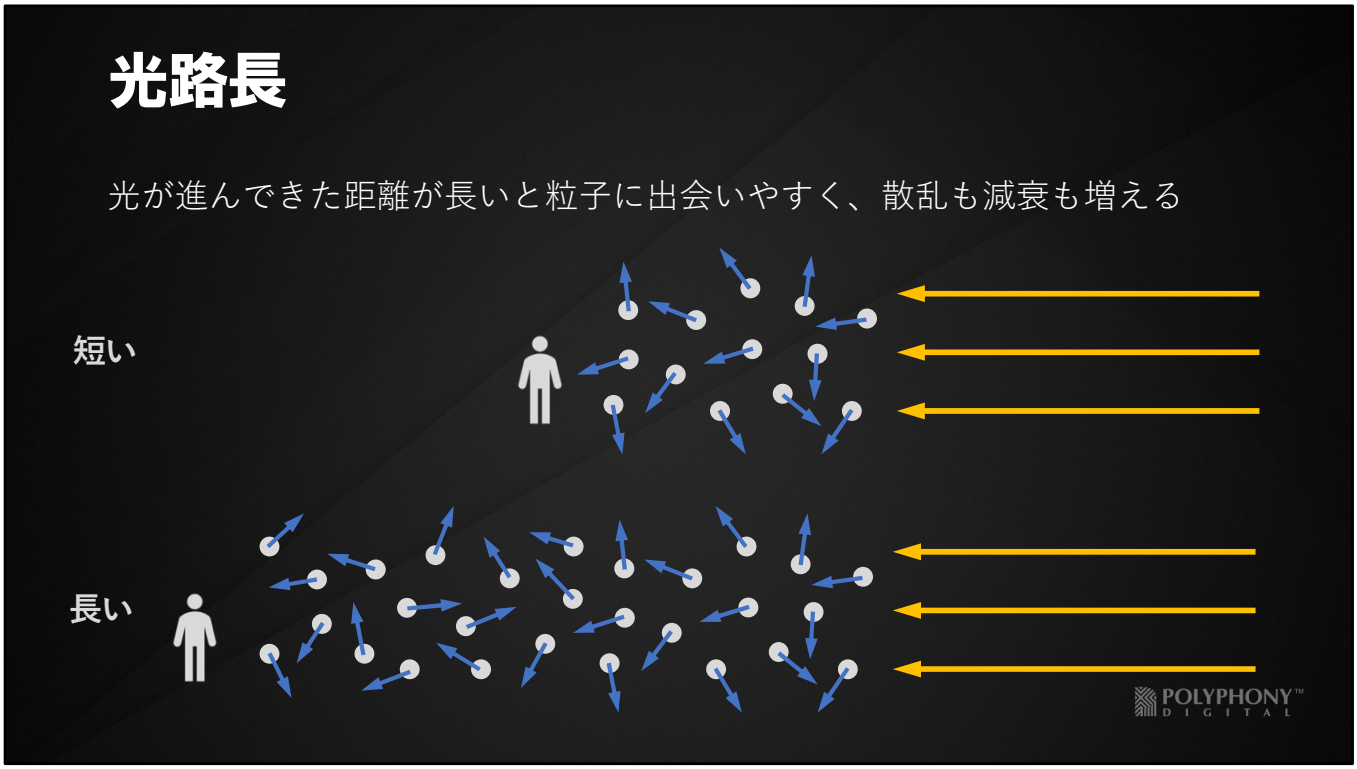
大気中には、小さな水滴や氷の粒の濃度が集中的に高い部分がある。そのミー散乱の光が**雲**として見えている

雲は何か異質な塊が浮いているように見えますが、その部分に散乱しやすい大きな粒子が集まっていて、ミー散乱が異常に多くなっています

# 粒子濃度

雲に影が出るのは、雲のミー散乱で太陽光が**減衰**するから

散乱が多いと、減衰も大きくなります。雲の下は太陽の光が届きにくくなり、
これが影のように見えます



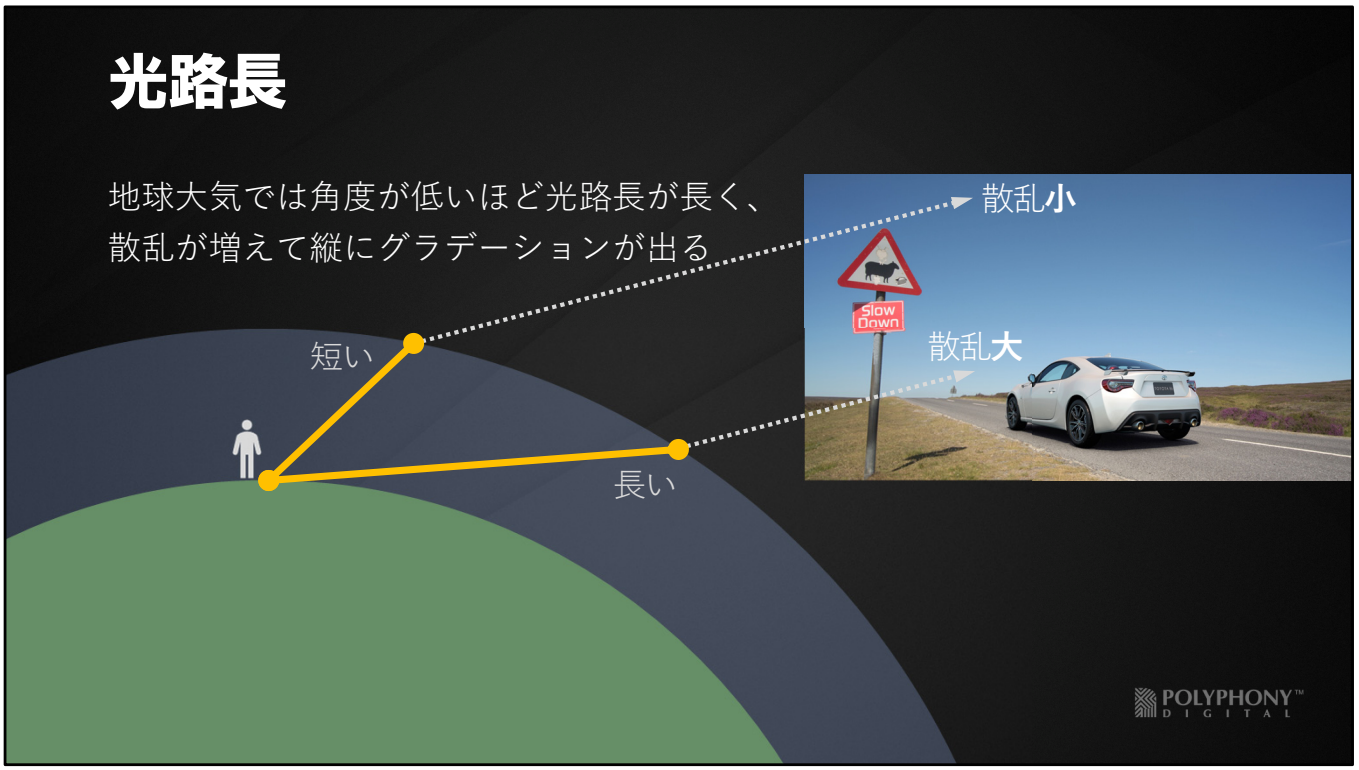

マクロで考えるときにもうひとつ大事なのは、距離です。これも直感に反しないと思います
濃度が同じでも、光が通る距離が長いほど散乱は増えます

# 光路長

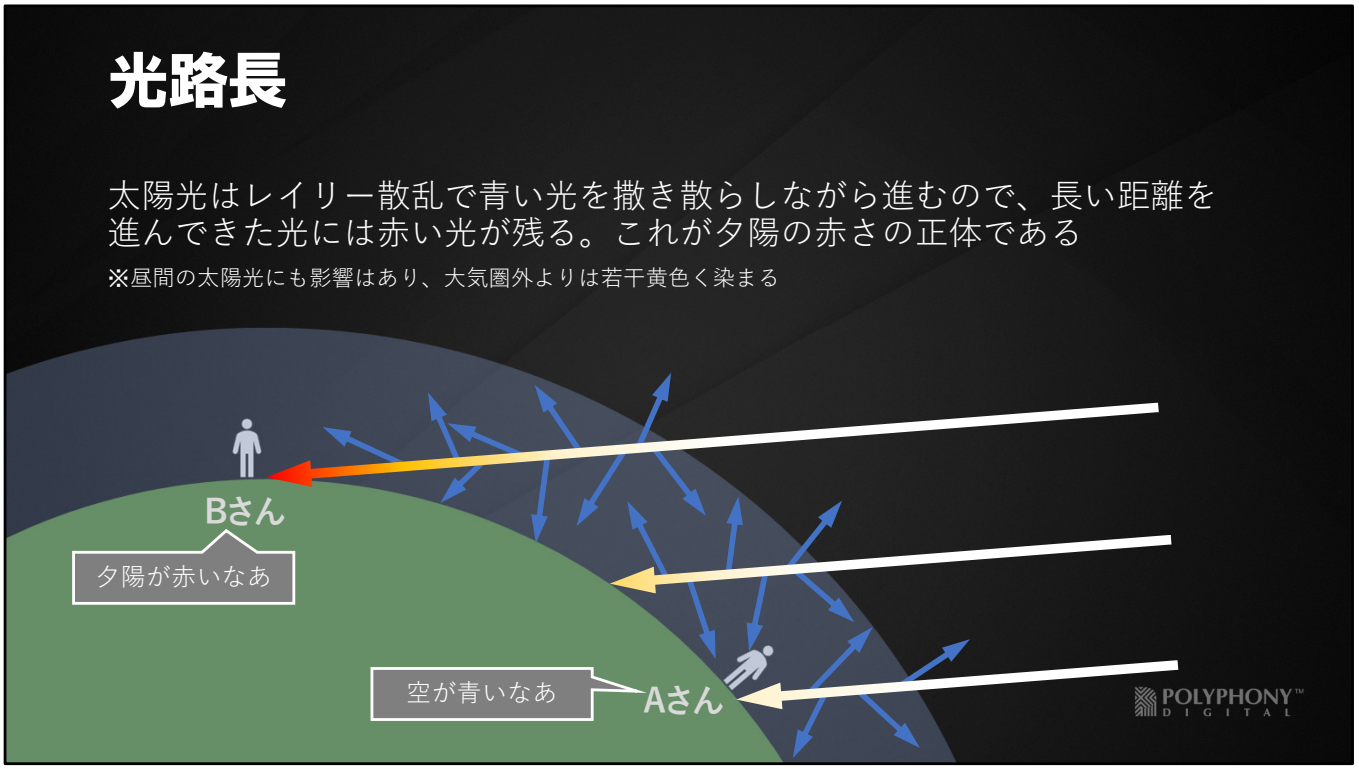
霞や霧は**光路長がとても短い空**(の一部)。散乱は空とまったく同じだが、
途中に山や建物がある場合はそれより手前の大気散乱しか見えないので
空の色が薄く乗ったような見え方になる

これはスモッグが出ているときの東京の写真です
遠いほど散乱が増えて、どんどん空に近くなっていくことがわかります

上から下まで大気全体が光っている、ということもわかりますね



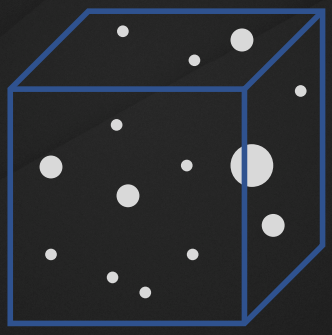

地球大気はこんなぐあいなので、上より横のほうが大気の距離が長くなっています
地球の半径は大きいので、実際は10倍以上違います




レイリー散乱が青い光を太陽光から奪っていくので、白い太陽光線は距離に応じて赤みを帯びるようになります
AさんとBさんが見ている空の違いは、光路長が違うだけです

# 散乱の法則(マクロ視点)

散乱は基本的にマクロで扱われ、**単位体積あたり**でまとめて考える



POLYPHONY DIGITAL

粒子ひとつの散乱の特徴、濃度、それから光路長。このあたりが散乱を特徴づけます

# 散乱の法則(マクロ視点)

マクロ視点では、〇〇断面積は粒子濃度込みで**単位体積あたりの断面積**となり、以下のように変化する

- 散乱しやすさ：**散乱係数** (scattering coefficient)
- 吸収しやすさ：**吸収係数** (absorption coefficient)

- **散乱係数+吸収係数= 減衰係数** (extinction coefficient)

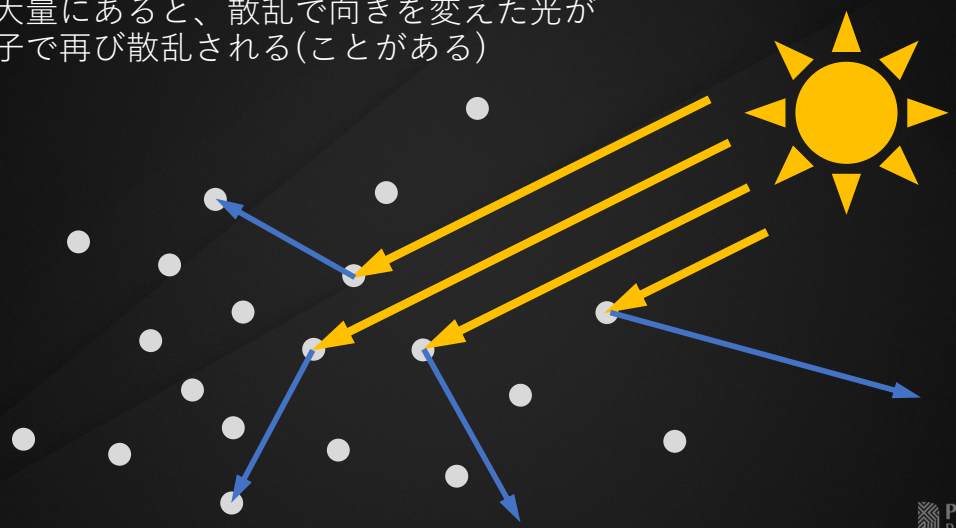
光が単位距離進むあたりの散乱率･吸収率･減衰率となって、微積分などが計算しやすいカタチに

POLYPHONY DIGITAL

体積あたりで考えると、用語はこうなります

# 多重散乱

粒子が大量にあると、散乱で向きを変えた光が
別の粒子で再び散乱される(ことがある)

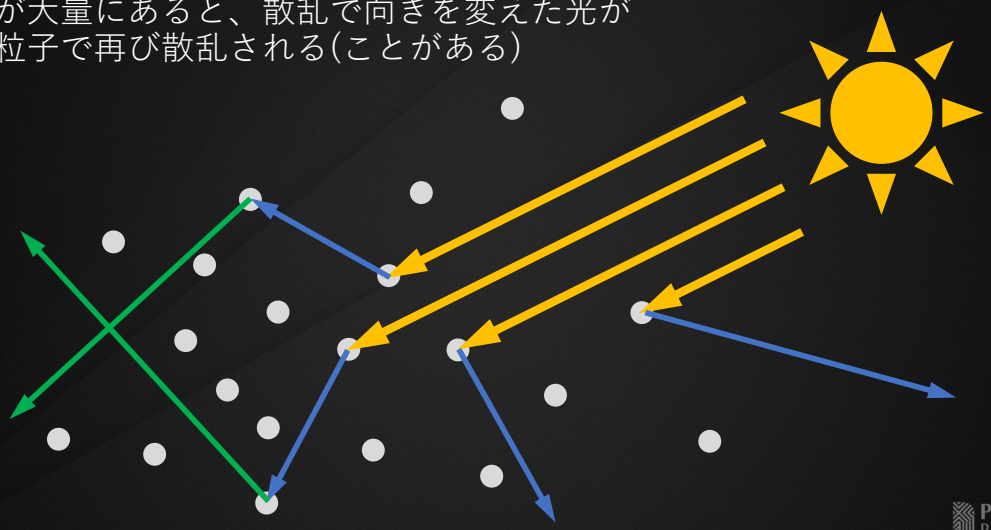


POLYPHONY DIGITAL

マクロで考えるときにもうひとつ無視できない現象が多重散乱です

# 多重散乱

粒子が大量にあると、散乱で向きを変えた光が
別の粒子で再び散乱される(ことがある)

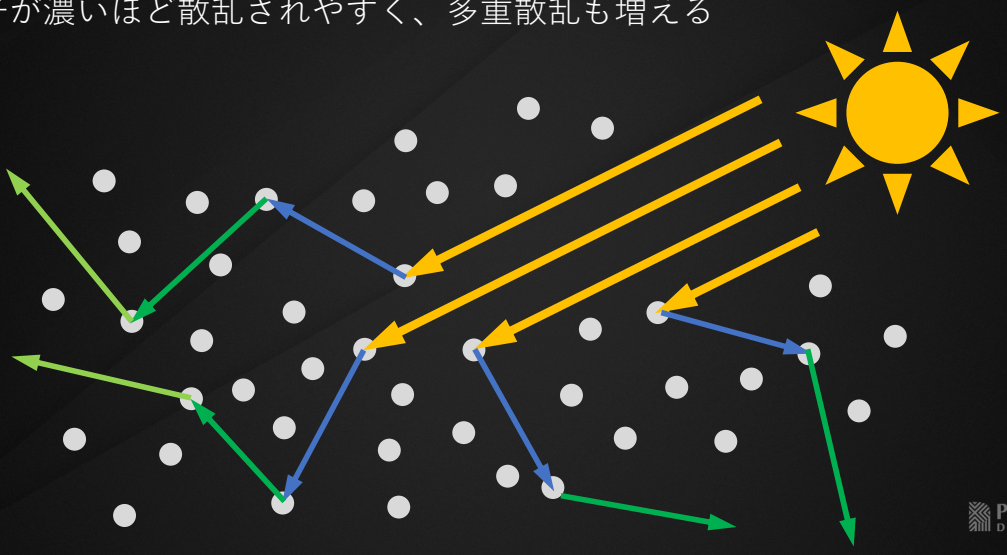


POLYPHONY DIGITAL

散乱された光がまた散乱される現象です。入射光の向きが変わるので、位相関数の向きも変わります

# 多重散乱

粒子が濃いほど散乱されやすく、多重散乱も増える




粒子が大きいとか、粒子が濃いとかで散乱しやすい空間では多重散乱が増えていきます

# 多重散乱

雲が白い塊に見えるのは、雲の内部で膨大な多重散乱が起きているため。
雲の散乱のうち**7〜8割は多重散乱**と言われる

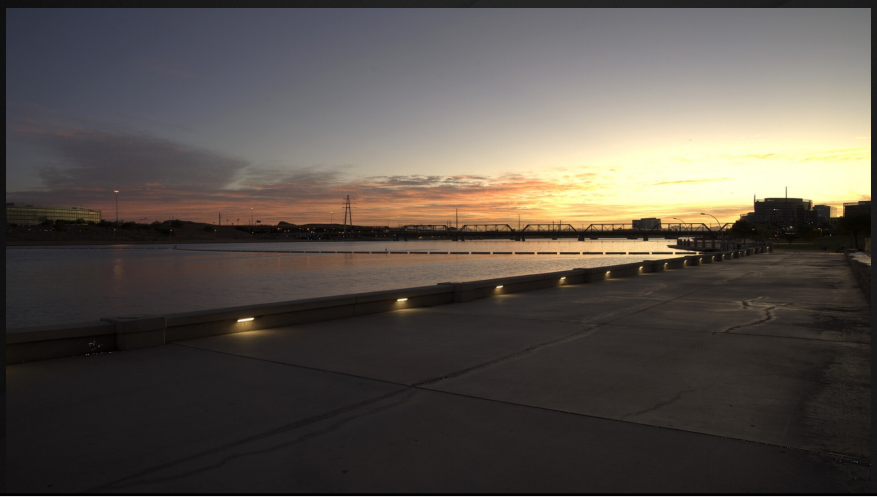
雲の白さ、柔らかい陰影は多重散乱によるところが大きいです

# 多重散乱

濃霧で太陽の方向がよくわからなくなるのは、大量の多重散乱で光が四方八方に散らされるため。雲の内部もだいたいこんな感じ

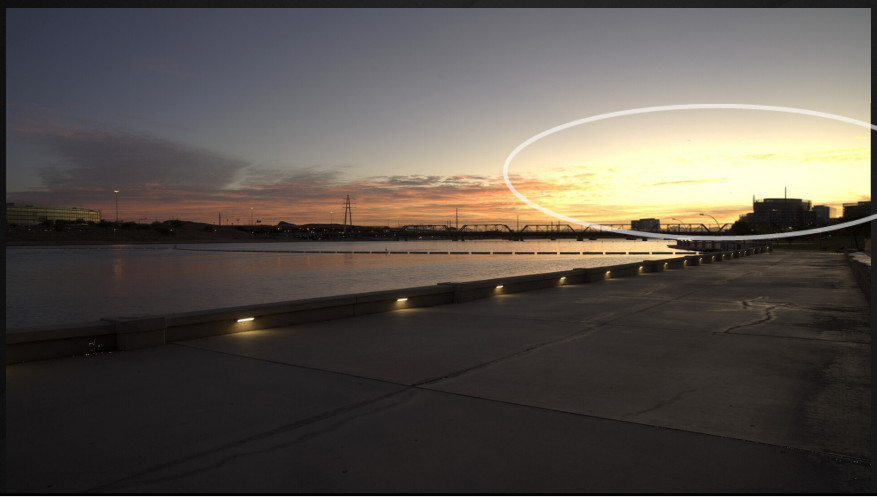
# 多重散乱

夕焼けの美しくオレンジ色に輝く空は、その大部分が長い光路長において発生した多重散乱による



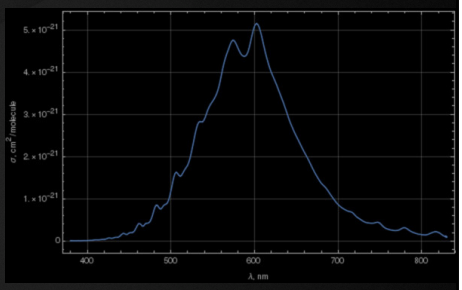
# 多重散乱

夕焼けの美しくオレンジ色に輝く空は、その大部分が長い光路長において発生した多重散乱による



# オゾン層の影響

話は変わるが、オゾン層にはChappuis bandという可視光吸収帯があり、夜明けや夕闇の、所謂**ブルーアワー**の深い青に影響している。意外と重要！




POLYPHONY DIGITAL

これは比較的最近注目されるようになりました。オゾン層が吸収するのは紫外線だけではありません

# ここまでのまとめ

空も雲も霞も霧も、ぜんぶ同じ**大気散乱**
粒子と濃度と光路長が違うだけ！

# 本日の内容

1. 空と雲の物理
2. **現代的な大気散乱シミュレーションと関連研究**
3. GT7における天球レンダリング手法(オフライン)
4. GT7における天球レンダリング手法(ランタイム)
5. まとめ

POLYPHONY DIGITAL

まず、空と雲の基本的な物理のお話をします

次に、空をCGでシミュレーションするさいの基本をお話しします

# 現代的な大気散乱シミュレーションと関連研究

POLYPHONY DIGITAL

# 考え方

大気散乱を**任意の光源**、**任意の視点**で物理的にレンダリングするのが大気散乱シミュレーション。

そのために**[ボリュームレンダリング方程式]**を解く

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

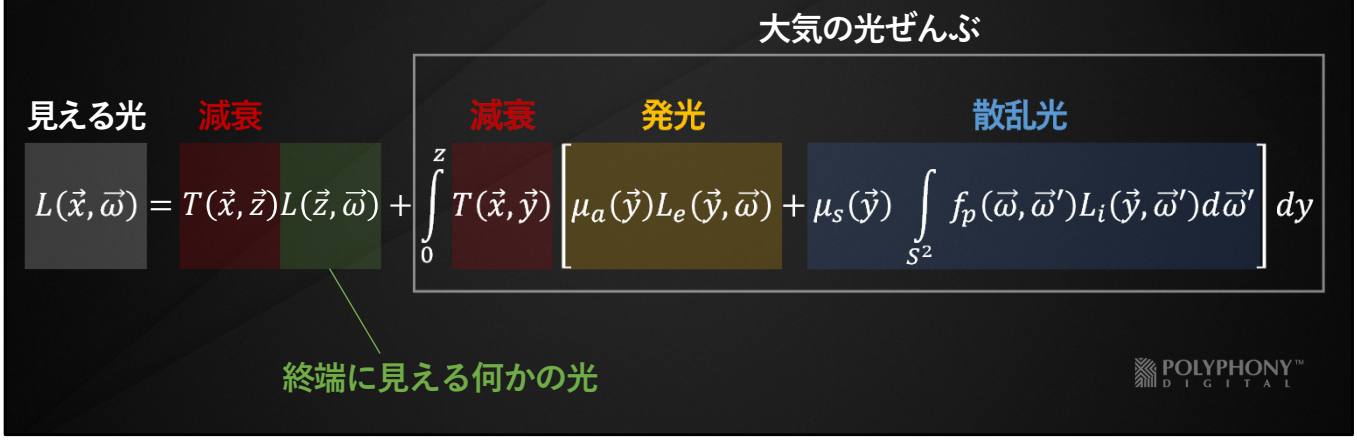
POLYPHONY DIGITAL

ボリュームレンダリング方程式というものがあります

えらい迫力のある式ですけど、これはたんに「考え方」を表しているだけなので、怖がらないでください

# ボリュームレンダリング方程式

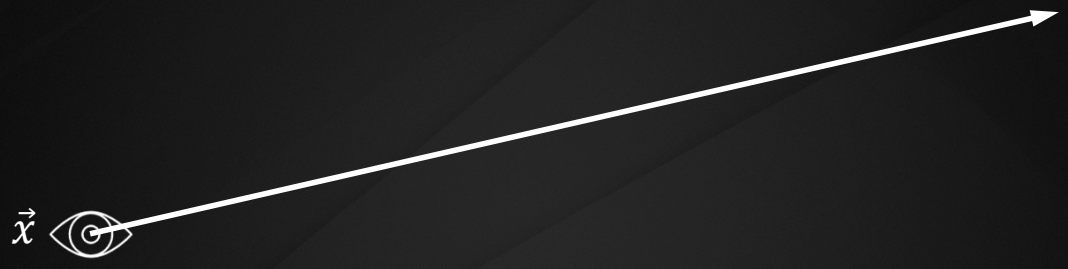
ざっくりと、こんな意味




ある方向をみたときに見える光は、その方向の大気の散乱光と発光ぜんぶと終端から入射する光とが大気を通って減衰してきたものの合計ですよ、という意味です

「終端に見える何か」ってのは、ちょっと前に出した東京の写真のビルとかです
途中に遮るものがあったら、その奥は見えないからその手前だけを考えればいいよ、ってことを表しています

# ボリュームレンダリング方程式




$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

POLYPHONY DIGITAL

座標xの位置から、ある方向を見るとします

この空間は大気で満たされているとします

# ボリュームレンダリング方程式

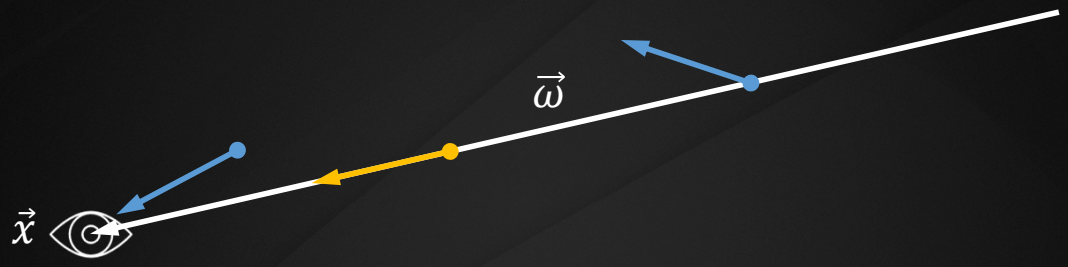
$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

POLYPHONY DIGITAL

この方向を見て見える光は、見た方向に平行に入ってくる光です

すごく小さな穴の開いた箱の中から外をのぞいているようなイメージ

# ボリュームレンダリング方程式




$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

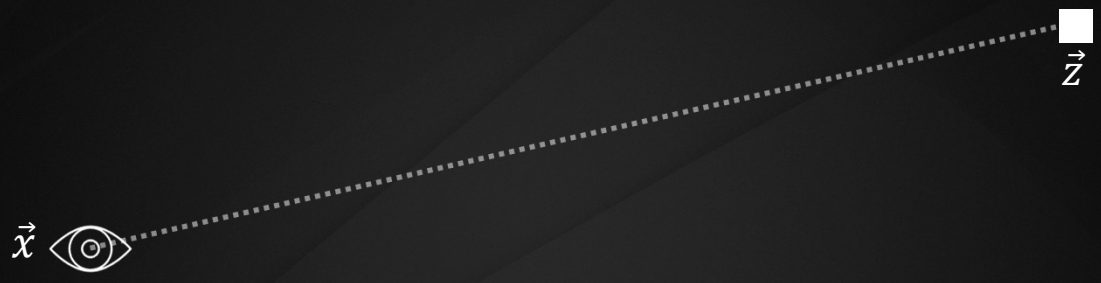
POLYPHONY DIGITAL

そのベクトルにぴったり重なる光「だけ」が見えます

この図だと、黄色い矢印の光だけが見えます
青い矢印の光は見えません

左の青い矢印の光は見えるんじゃね、って思うかもしれませんが、見えません
その方向は見ていないから。そういう世界です

# ボリュームレンダリング方程式



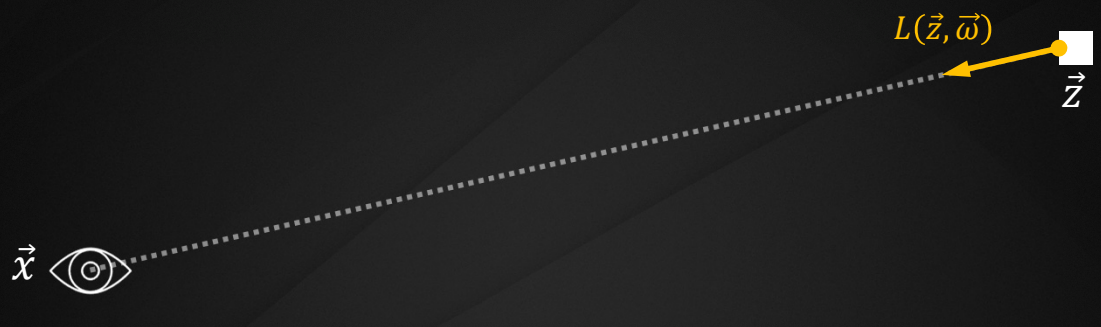

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

POLYPHONY DIGITAL

座標zに壁があるとしますよ

壁より向こうは見えません

# ボリュームレンダリング方程式



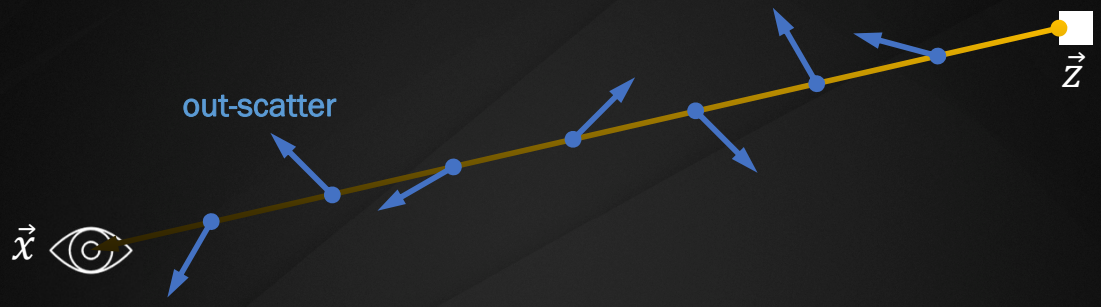

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

終端輝度

POLYPHONY DIGITAL

壁が太陽に照らされたりとかして、視点方向にも光が出ます

# ボリュームレンダリング方程式




$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度

POLYPHONY DIGITAL

ところが、座標xと座標zの間にある大気の散乱と吸収によって、この光が視点まで進む間にどんどん光がいなくなって減衰します
散乱で光が逸れて出ていくことを out-scatter といいます

# ボリュームレンダリング方程式

$$L(\vec{x},\vec{\omega}) = \underline{T(\vec{x},\vec{z})}\,\underline{L(\vec{z},\vec{\omega})} + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度

POLYPHONY DIGITAL

これで右辺のいちばん左の項は終わりです。簡単でしょう？

# ボリュームレンダリング方程式

減衰関数$T$について

- 2点間の**光の減衰**をあらわす関数

$$T(\vec{x},\vec{y}) = \exp\left(-\int_0^y \sigma_t(\vec{x}-s\cdot\omega)ds\right)$$

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　減衰関数

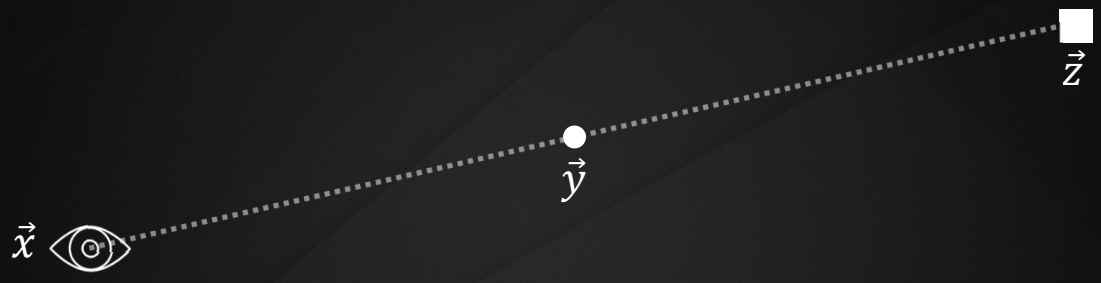
POLYPHONY DIGITAL

さて、ここで出てきたTについて説明します

これは前に説明した、散乱と吸収による光の減衰をあらわす専用の関数です
散乱で減衰しているのになんで散乱と別なの？と思うかもしれませんが、式を整理すると散乱光の積分の外に出るんです

このことはシミュレーションをするにあたって有利にはたらきます

# ボリュームレンダリング方程式



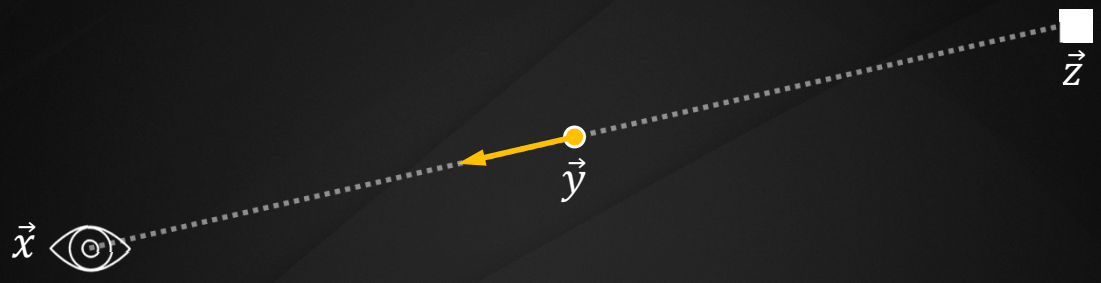

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度

POLYPHONY DIGITAL

次。大気中の点yを考えます

# ボリュームレンダリング方程式

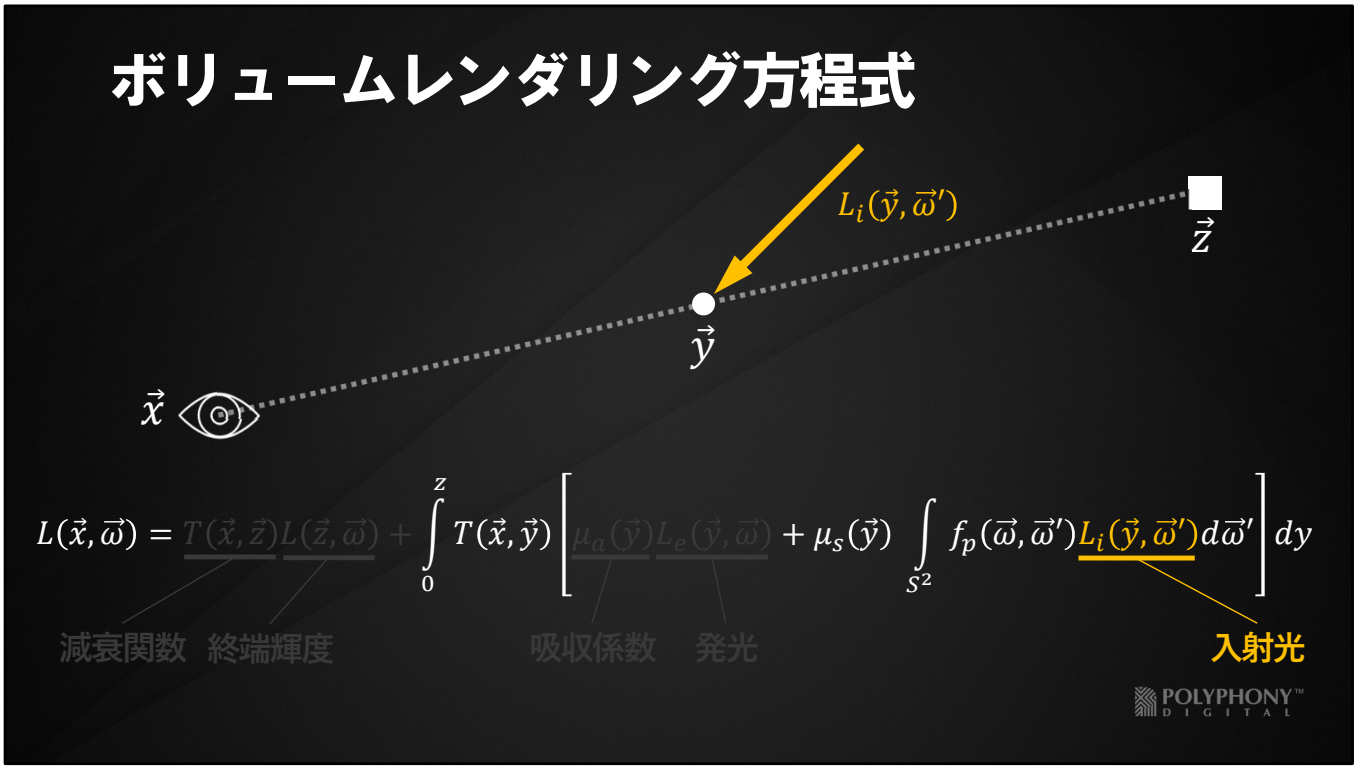



$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

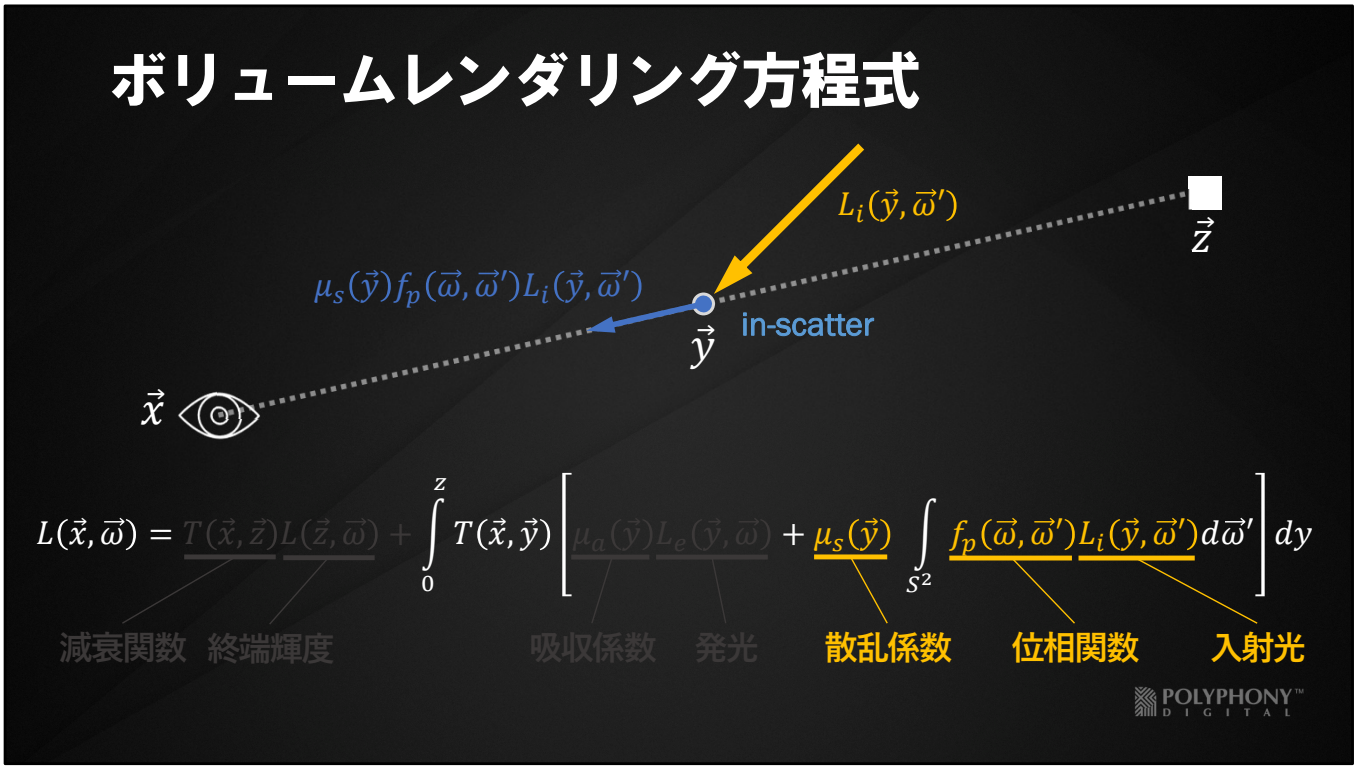
減衰関数　終端輝度　吸収係数　発光

POLYPHONY DIGITAL

地球大気では通常考えなくてもよいですが、大気が自分自身で発光している場合、その一部は吸収を受けつつ視点方向に出ます
オーロラや大気光は大気の発光です

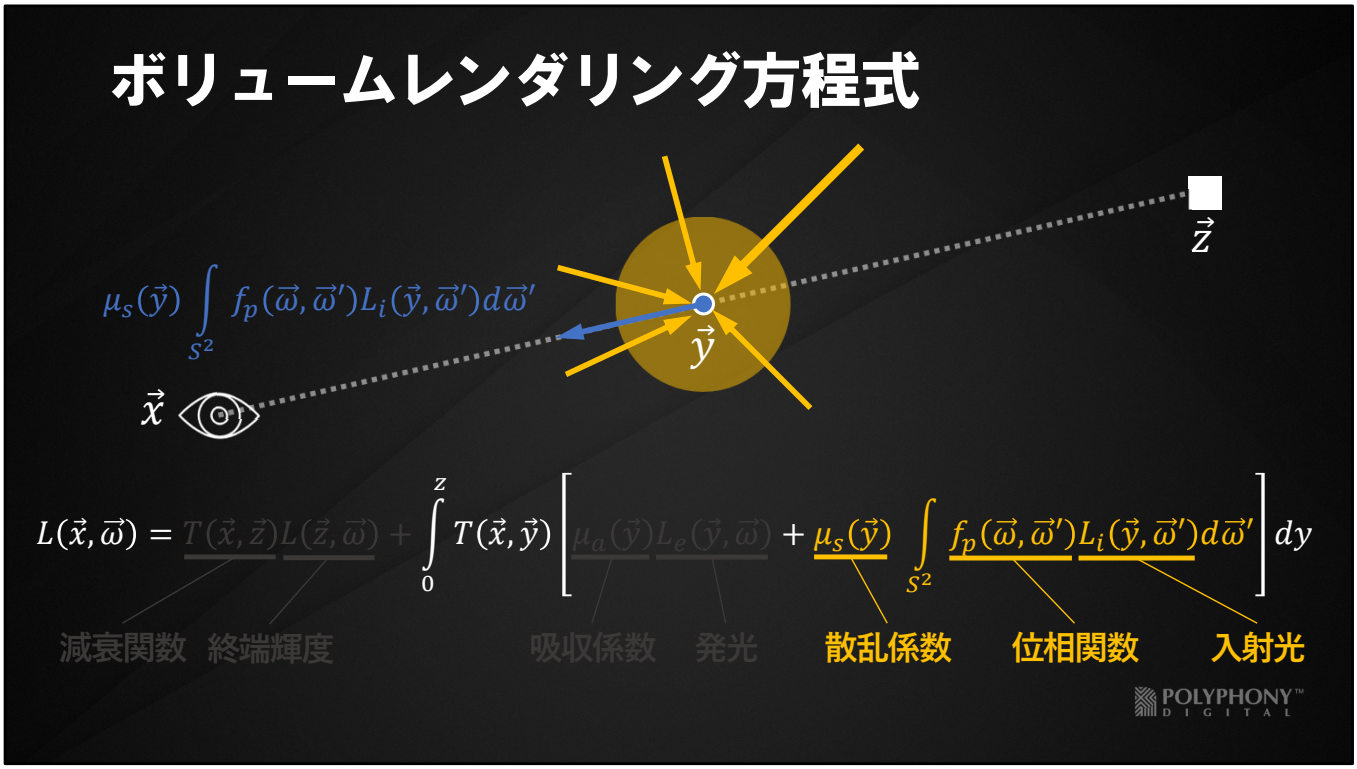



ここからです。この点は光をうけます。たとえば太陽光



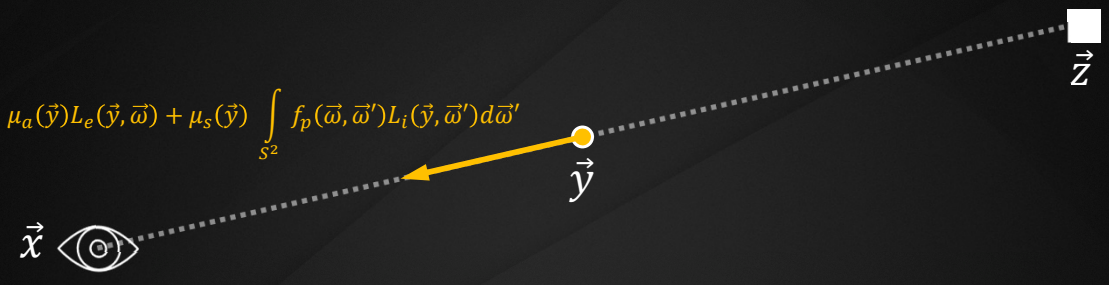

入射した太陽光は散乱係数と位相関数にしたがって散乱し、その一部は視点方向にむかいます

視点方向に向いていない光が向きを変え、視点方向に向かいます。つまり光が増えます
これを in-scatter といいます




空中ですから、光は一方向からの太陽光だけじゃなく全周からやってきますね。どの方向からきた光も同じように、散乱光の一部が視線方向に向かいます
全球積分すると、この点の in-scatter の総量がわかります

# ボリュームレンダリング方程式



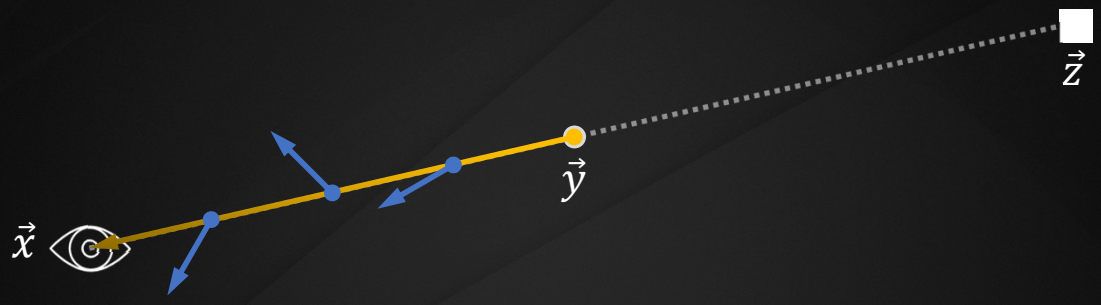

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　吸収係数　発光　散乱係数　位相関数　入射光

POLYPHONY DIGITAL

自発光と散乱光を合計して、大気中のこの点が視点方向に出す光がわかりました

# ボリュームレンダリング方程式



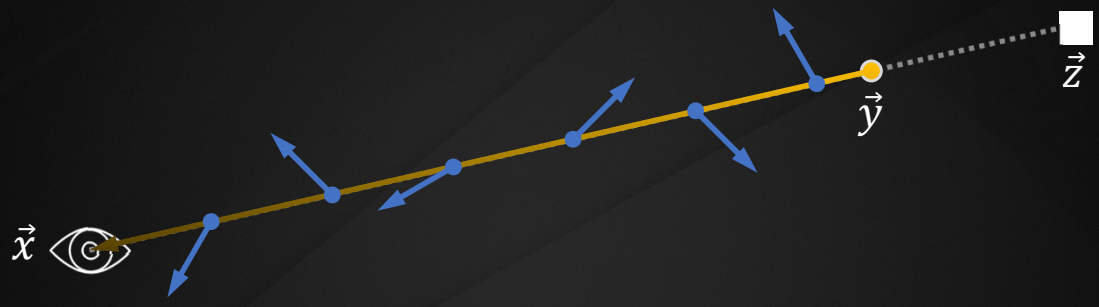

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　減衰関数　吸収係数　発光　散乱係数　位相関数　入射光

POLYPHONY DIGITAL

で、さっきと同じように、この光も視点まで進む間に out-scatter と吸収で減衰します

# ボリュームレンダリング方程式



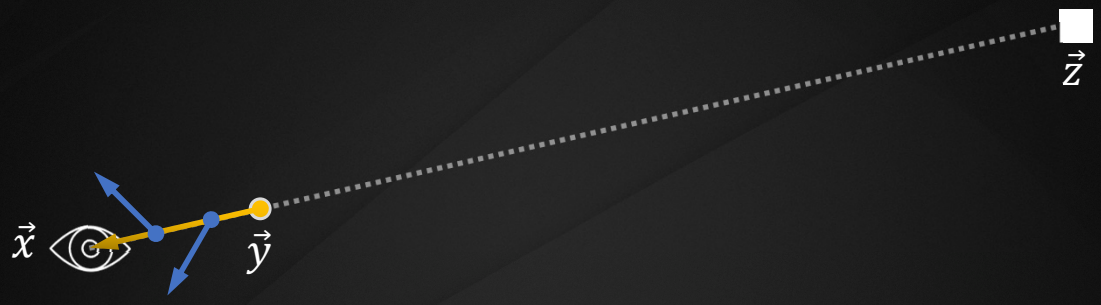

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　減衰関数　吸収係数　発光　散乱係数　位相関数　入射光

POLYPHONY DIGITAL

さて、いまの話。yがどこにあっても成立しますよね？
ここでも

# ボリュームレンダリング方程式



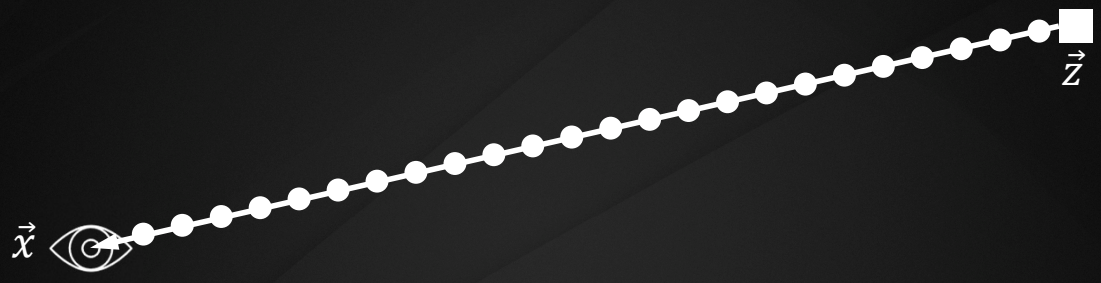

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　減衰関数　吸収係数　発光　散乱係数　位相関数　入射光

POLYPHONY DIGITAL

さて、いまの話。yが大気中のどこにあっても成立しますよね？
ここでも

# ボリュームレンダリング方程式



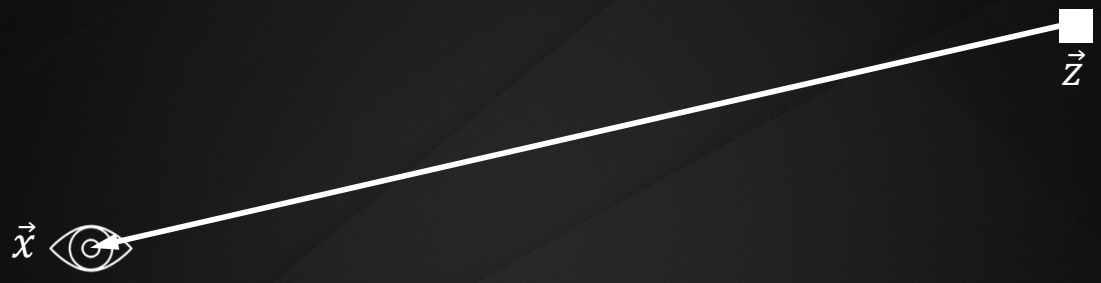

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　減衰関数　吸収係数　発光　散乱係数　位相関数　入射光

POLYPHONY DIGITAL

ですから、終端から視点までの全区間から届く光を積分して集めてやれば、この区間の大気から届く光の総量がわかります

# ボリュームレンダリング方程式

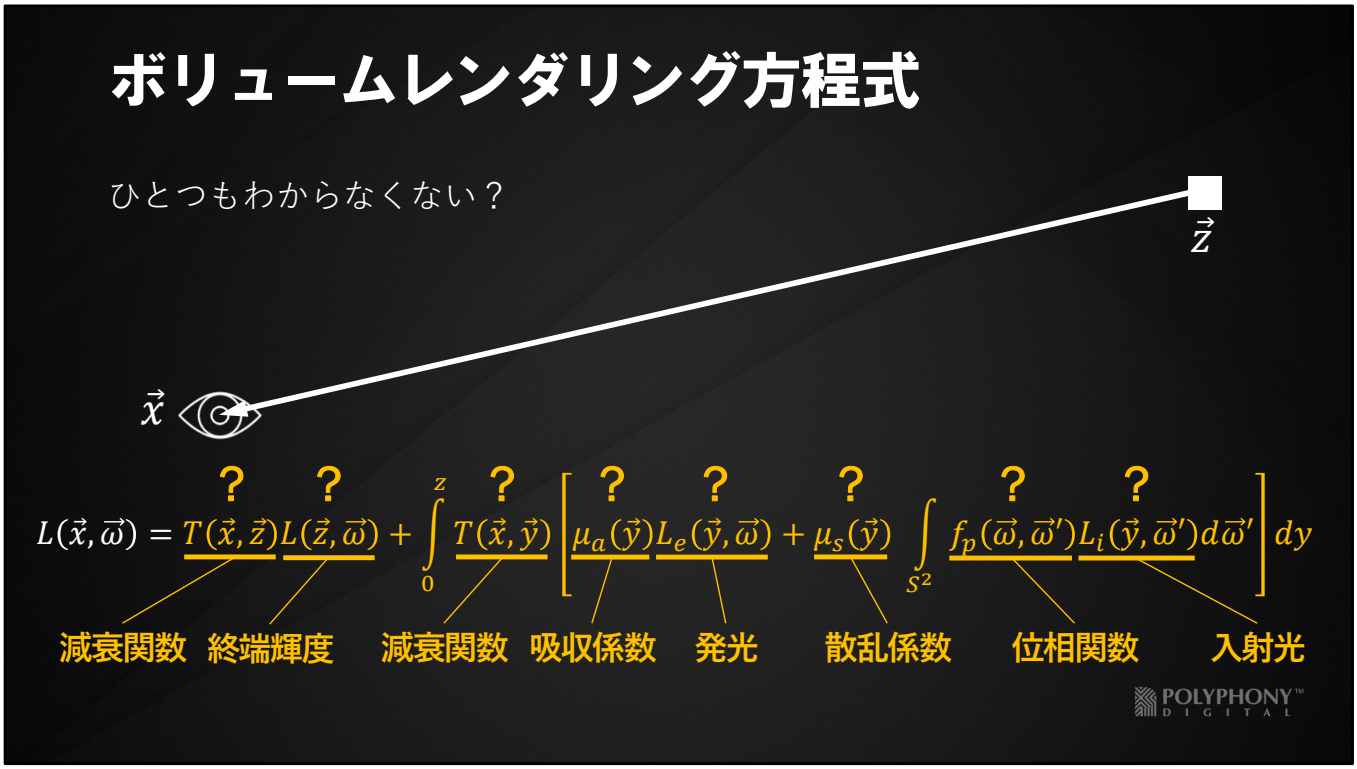



$$L(\vec{x},\vec{\omega}) = \underline{T(\vec{x},\vec{z})}\,\underline{L(\vec{z},\vec{\omega})} + \int_0^z \underline{T(\vec{x},\vec{y})}\left[\underline{\mu_a(\vec{y})}\,\underline{L_e(\vec{y},\vec{\omega})} + \underline{\mu_s(\vec{y})}\int_{S^2}\underline{f_p(\vec{\omega},\vec{\omega}')}\,\underline{L_i(\vec{y},\vec{\omega}')}\,d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　減衰関数　吸収係数　発光　散乱係数　位相関数　入射光

POLYPHONY DIGITAL

これがボリュームレンダリング方程式の全貌です




でも、各点の大気がどんなで、どんな光がくるかとか、減衰関数がどんなだとか、わからないことだらけですよね

レンダリング方程式ってのはそういうもので、理屈はこうだから内容はお前が決めろ、という手厳しいものです
ですから、なんとなく雰囲気でわかればよいです。解く人だけが苦労します

# ボリュームレンダリング方程式

ボリュームレンダリング方程式を解くには、任意の座標における散乱係数・吸収係数・位相関数、ライティングが必要になる

これはすなわち**地球大気をモデリング**することである

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

減衰関数　終端輝度　減衰関数　吸収係数　発光　散乱係数　位相関数　入射光

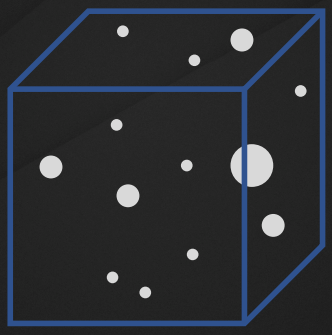
大気モデリングだけで決まる要素

POLYPHONY DIGITAL

しかし、大気のありようで大部分は決まっちゃいます。入射光も空からの散乱光が少なくないので、大気のありようと密接に絡みます
だから大気のモデリングがとても重要です

# 大気のモデリング

地球大気には多様な粒子が多様な分布で浮いている。大気のモデリングが現実の地球大気に近いほどリアルなシミュレーション結果が期待できるが、複雑であればあるほど方程式を解くのも難しくなる



POLYPHONY DIGITAL

モデリングが本物に近いほどリアルになります。これはふつうのCGモデリングと変わりないです

# 大気のモデリング

**大気分子+α**(レイリー散乱)・**エアロゾル**(ミー散乱)に分けて、鉛直方向の密度変化のみモデリングするのが一般的。**指数関数減衰**がよく使われる。**散乱係数・減衰関数**を定義することに等しい

- **大気分子**
  組成や濃度分布は地球上いつでもどこでもおおむね安定していて、単純化しやすい
- **エアロゾル**
  組成や濃度分布は常時不定かつ複雑。地域や時間帯でも異なり、正確な再現は難しい。大胆に単純化されることが多く、どちらかというと係数を変化させて空のムードを変えることに主眼が置かれる

POLYPHONY DIGITAL

大気分子とエアロゾルに分けてモデリングするのが一般的です
鉛直方向のみをモデリングするのは、対称性を高めて計算をシンプルにする狙いがあります

大気分子は比較的簡単で、エアロゾルは難しいです

# 減衰関数のモデリング

**減衰関数**はさまざまな計算で使用頻度が高く(方程式にも2回出てくる)、CG業界で研究が盛んな分野である

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y})\int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

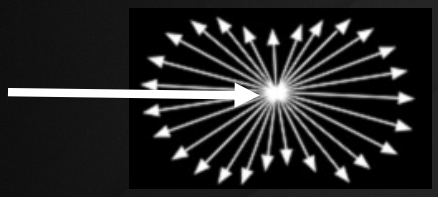
POLYPHONY DIGITAL

ボリュームレンダリング方程式のところでも説明したように、減衰関数はシミュレーションの重要なピースのひとつです
レイマーチングやRatio Tracking等で事前計算したり、さまざまな工夫があるところです

# 位相関数のモデリング

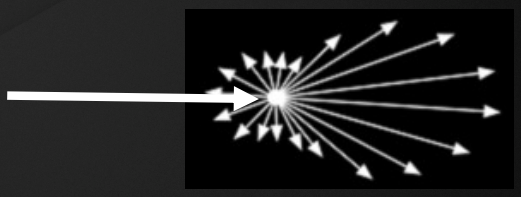
大気分子・エアロゾルそれぞれにひとつずつ直接定義されることが多い

**レイリー散乱**
主に**大気分子**



- 組成が既知で粒径分布も安定
- 位相関数に**解析解がある**

**ミー散乱**
主に**エアロゾル**

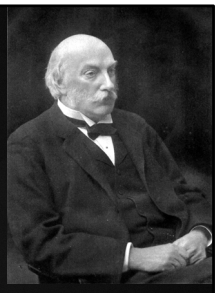


- 組成は不定で複雑、粒径分布もさまざま
- 位相関数に**解析解が無い**

POLYPHONY™ DIGITAL

エアロゾルが起こすミー散乱の位相関数は場所によってかなり変わりますが、
たいていはひとつで代表します

# レイリー散乱の位相関数



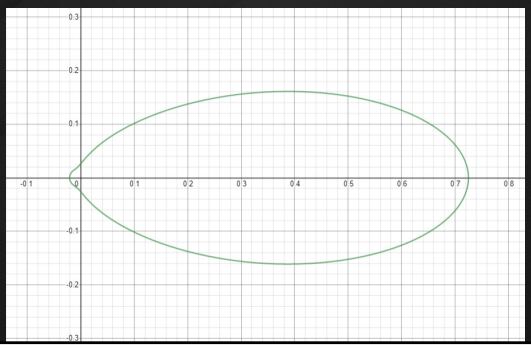
- 大気分子の組成は[**窒素**78%、**酸素**21%、**その他**1%]なので、窒素と酸素の位相関数を近似できればほとんど終了
- シンプルで解析解があるので解きやすく、大きな誤差も出にくい

POLYPHONY™ DIGITAL

レイリー散乱の位相関数はいつも変わりません。ここで誤差を出しているシミュレーションはほとんどないと思います

# ミー散乱の位相関数

- ミー散乱粒子は**土地の気候**や**その日の天気**、**時刻**でも内容が大きく変わるので、組成も粒径も**不定**
- 散乱光に位相差があり、干渉によって非常に複雑になる
- Henyey-Greenstein, Cornette-Shanks等のな近似関数がしばしば使われるが、精度はあまり良くない



POLYPHONY™ DIGITAL

ミー散乱は再現が難しいし、そもそも現実がどうなっているかもあまりよく知られていないので、割り切ったつくりが多いです

# 雲について

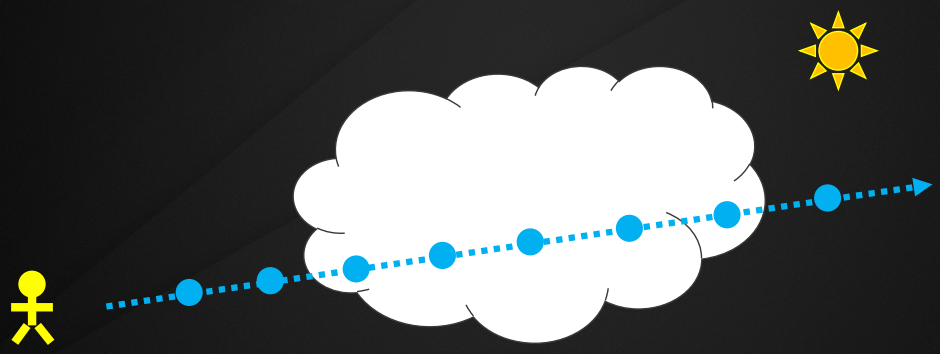
雲は以下のような特徴があるので、ゲームCGにおいては空と分けて**レイマーチング法**でレンダリングされることが多い

- 雲粒はほぼ水だけで、組成が一定
- 大量の多重散乱を伴うミー散乱のみで透明度が低く、誤差がわかりにくい
- 出たり消えたり形を変えたり流れたり、高速に変化してほしい

POLYPHONY™ DIGITAL

# レイマーチング法

- 空のレンダリング同様、ボリュームレンダリング方程式に基づいて散乱計算を行う。ただし**ランタイムで行う**。
- 一般には、何らかのノイズによって定義された雲の密度場に対してレイマーチングを行うことで計算される。



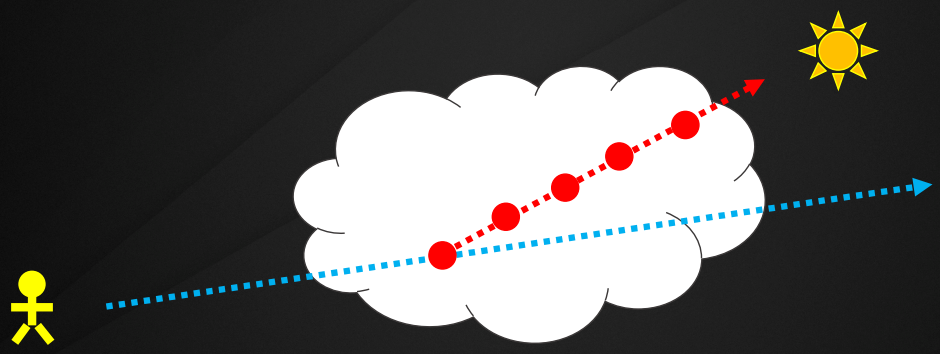
カメラからのレイマーチング

POLYPHONY DIGITAL

レイマーチングはボリュームレンダリングに適した手法です

# レイマーチング法

- 空のレンダリング同様、ボリュームレンダリング方程式に基づいて散乱計算を行う。ただし**ランタイムで行う**。
- 一般には、何らかのノイズによって定義された雲の密度場に対してレイマーチングを行うことで計算される。



太陽へのレイマーチング（寄与計算）

# 多重散乱のモデリング

- 多重散乱はGlobal Illuminationであり、大気全域で正しく計算するには長時間かけてパストレーシングするしかないが、無視すると薄暗くて単調な空となる。雲は黒くなってしまう

画像：Computer Graphics II: Rendering CSE 168[Spr 20],Lecture 15: Volumetric Rendering Ravi Ramamoorthi

# 多重散乱のモデリング

完璧に再現するのは困難だが、現代では

パストレーシングで得たリファレンスに解析的な関数をフィッティングさせたり、LUTを使ったりして光を加える

多重散乱はミー散乱のピーキーで複雑な位相関数を結果的に畳かすことになるので、多重散乱込みなら位相関数の精度が悪くてもある程度成り立つという事情もあり、雲のレンダリングでは特に有効にはたらく

POLYPHONY™ DIGITAL

# 代表的な研究（空）

1993, 1996　西田らによる空モデル

CG分野における、物理的な空のレンダリング論文として今なお基礎となる研究。[Nishita et al. 1993, 1996]

2008　Precomputed Atmospheric Scattering

コンパクトなLUT上で大気による光散乱を計算することで高速に空をレンダリング。[Bruneton and Neyret 2008]

2012　An Analytic Model for Full Spectral Sky-Dome Radiance

事前に長時間かけて散乱を計算しておき、解析的な関数にフィッティング。[Hosek and Wilkie 2012]

2020　A Scalable and Production Ready Sky and Atmosphere Rendering Technique

新しいLUTと多重散乱計算方法を提案し、メモリと計算コストの削減を実現。[Hillaire 2020]

2021　A Fitted Radiance and Attenuation Model for Realistic Atmospheres

[Hosek and Wilkie 2012]の発展版。計測ベースの大気モデルを用いるなど、多くの拡張を含む。[Wilkie et al. 2021]

# 代表的な研究（雲） [ゲーム、リアルタイムレンダリング]

- 2015, 2017　Nubis: Authoring Real-Time Volumetric Cloudscapes with the Decima Engine
  - Horizon: Zero Dawnで用いられたリアルタイム雲レンダリング手法。ノイズベースの雲モデリングとレイマーチングによるライティング。 [Schneider 2015, 2017]
- 2016　Physically Based Sky, Atmosphere & Cloud Rendering in Frostbite
  - Frostbiteにおけるリアルタイムの空・雲レンダリング手法[Hillaire 2016] 。[Schneider 2016]をベースにしている。
- 2019　Creating the Atmospheric World of Red Dead Redemption 2: A Complete and Integrated Solution
  - Red Dead Redemption 2で用いられたリアルタイム雲レンダリング手法[Bauer 2019] 。[Schneider 2015, 2017]をベースに、様々な改良が加えられている。

**日本における発表**

- 2017　『グランブルーファンタジー Project Re:LINK』の“雲”表現にかける並々ならぬこだわり [GCC]
- 2018　ゼノブレイド２のレイマーチングを使った雲表現 [CEDEC]
- 2019　エースコンバット7における”空の革新”について [CEDEC]

POLYPHONY DIGITAL

# 本日の内容

1. 空と雲の物理
2. 現代的な大気散乱シミュレーションと関連研究
3. **GT7における天球レンダリング手法(オフライン)**
4. GT7における天球レンダリング手法(ランタイム)
5. まとめ

まず、空と雲の基本的な物理のお話をします

次に、空をCGでシミュレーションするさいの基本をお話しします

# GT7の天球レンダリング手法
## （オフライン）

※オフライン：ゲーム中のリアルタイム計算ではなく事前に時間をかけて計算する処理

POLYPHONY DIGITAL

# 動機

まず既存の大気散乱シミュレーションモデルをいくつか検討してみた

夕日がボンヤリしていて、どうも物足りない。**※GTは夕日に強いこだわりがある**

# 動機

そもそも論文で**Ground Truth**とされているパストレーシング画像の時点で満足できる夕陽が**ひとつもない！**

これは大気モデリング精度に問題があるのではないかと考え、リファレンスとなる大気散乱シミュレーションを独自に開発することを決断

POLYPHONY™ DIGITAL

# 目標

- 学術分野の大気観測データをベースとして、まず**計算コストは度外視して**再現精度を可能な限り追求する
- **空気分子のレイリー散乱**は昔から十分な精度で再現されていることが既にわかっているので、難があるのは再現が難しい**エアロゾルのミー散乱**
- ここにターゲットを絞ってサーベイを開始

POLYPHONY DIGITAL

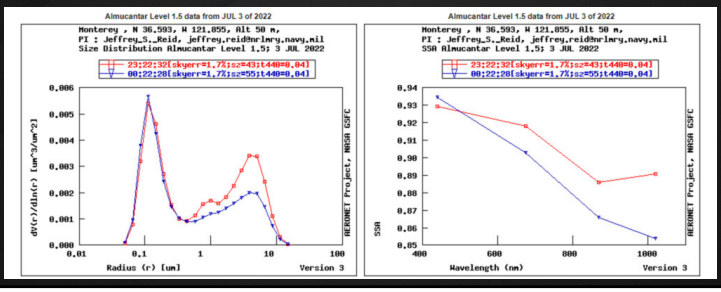
ちょっとお話しておきたいのは、実はこの時点の私は、レイリー散乱とミー散乱を知っている程度の素人だったということです

# サーベイ結果

- サーベイの結果、現実のエアロゾルの**位相関数・鉛直濃度分布**は一般的な大気散乱シミュレーションで使われる近似とはかなり異なっていた

- **AERONET**(**AE**rosol **RO**botic **NET**work, NASA提供)という、世界各地のエアロゾル観測データを照会できるネットワークがあり、たいへん役に立った

https://aeronet.gsfc.nasa.gov/


Almucantar Level 1.5 data from JUL 3 of 2022
Monterey , N 36.593, W 121.855, Alt 50 m,
PI : Jeffrey_S._Reid, jeffrey.reid@nrlmry.navy.mil
Size Distribution Almucantar Level 1.5; 3 JUL 2022
23:22:32[skyerr=1.7%;sz=43;t440=0.04]
00:22:28[skyerr=1.7%;sz=55;t440=0.04]
dV(r)/dln(r) [um^3/um^2]
Radius (r) [um]
Version 3
AERONET Project, NASA GSFC
Almucantar Level 1.5 data from JUL 3 of 2022
Monterey , N 36.593, W 121.855, Alt 50 m,
PI : Jeffrey_S._Reid, jeffrey.reid@nrlmry.navy.mil
SSA Almucantar Level 1.5; 3 JUL 2022
23:22:32[skyerr=1.7%;sz=43;t440=0.04]
00:22:28[skyerr=1.7%;sz=55;t440=0.04]
SSA
Wavelength (nm)
Version 3
AERONET Project, NASA GSFC

# 方針

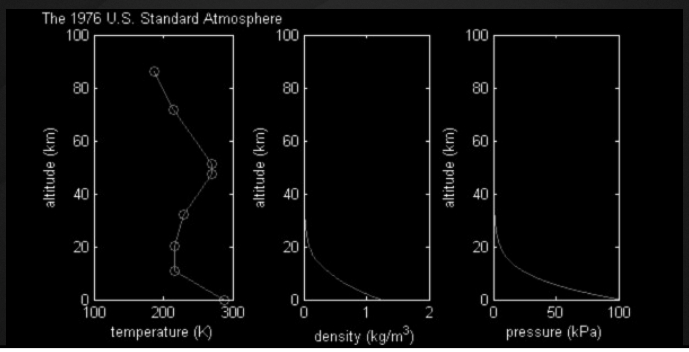
- エアロゾル観測は地上や空中、衛星から測定した大気散乱光をもとに大気の状態を推定しているので、推定された大気モデルを再現して散乱を逆演算すればリアルな空の色が復元されるはず！(割と力技)

- というわけで、**学術的な大気観測データをそのまま扱える**レンダラを開発することに

POLYPHONY DIGITAL

# 大気モデリング：空気分子

- U.S.Standard(1976)を引用
- 緯度経度や日時をパラメータに持つNRLMSISE-00という天文向けの高精度モデルもあったが、 比較検討した結果、**見た目は**ほとんど差が無かったのでシンプルなU.S.Standardを採用



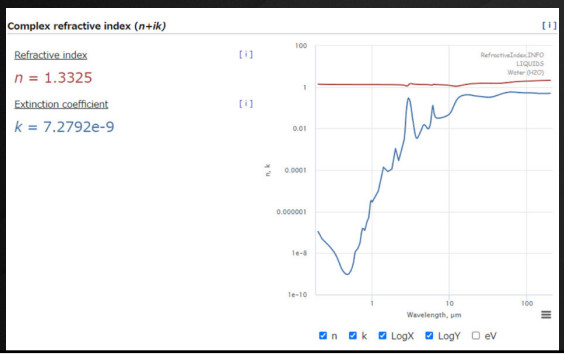

いちばん差が出る条件で3割くらい違ったが、見た目は水平線をよーく見比べてやっとわかる程度だった

[図:Wikipedia英語版 U.S._Standard_Atmosphere]

# 大気モデリング：エアロゾル粒子

## 物性は複素屈折率で定義

- 後述の**粒径分布**の定義と併せて、波長依存の**散乱係数・吸収係数・位相関数**が一発で決まる。入力がシンプルで便利
- 複素屈折率がわからないと話にならないが、気候変動がらみの研究のおかげで代表的なエアロゾル粒子はほとんど既知であり、探せば簡単に見付かる



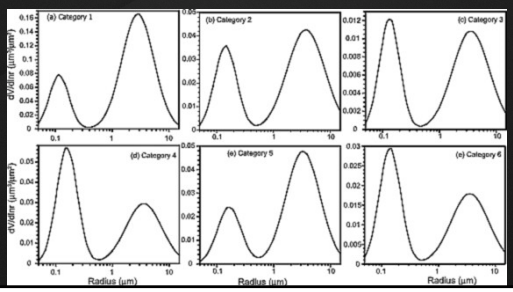

ここからはミー位相関数へのこだわりです
[図：RefractiveIndex.INFO]

# 大気モデリング：エアロゾル粒子

**粒径は対数正規分布で複数を定義**

- 気象分野ではこの近似がデファクトスタンダード
- ふたつのピークがあるのが特徴で、**バイモーダル分布**と呼ばれる
  小さいほうの粒径ピークはレイリー散乱に近く、若干の波長依存性を呈示する

- **この粒径分布の再現が本手法の最大の特徴**




POLYPHONY DIGITAL

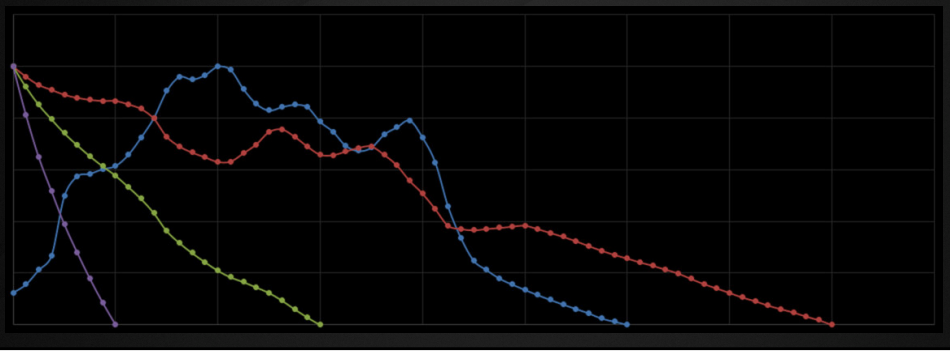
位相関数ひとつじゃぜんぜん足りない、ということがわかりました

[図:Omar 2005:Development of global aerosol models using cluster analysis of Aerosol Robotic Network (AERONET) measurements]

# 大気モデリング：エアロゾル粒子

**鉛直濃度分布はテーブルの和で定義**

- 論文のグラフをそっくりトレースできる。多くのデータをテストした
- 粒径分布のふたつのピークは異なるプロファイルを持つ。**粒径分布が高度で変化する**のも既存手法との大きな違い
- **大気の一部として雲をつくる**こともできる



POLYPHONY DIGITAL

# 大気モデリング：エアロゾル粒子

大気観測データの再現にはまったく不足ない書式が完成

POLYPHONY DIGITAL

# skysim

- 設定された大気データに基づいてボリュームレンダリング方程式を計算し、空をレンダリングするためのオフラインレンダラ skysim を開発。
  - アルゴリズムとしてボリュームのパストレーシングを用いる。
  - C++で2万行弱、開発は他の作業もしながら1年程度。

- 様々な特徴
  - フルスペクトルパストレーシング
  - 太陽光スペクトル
  - 地面色の設定
  - XYZ等色関数
  - Mie散乱位相関数
  - 最適化

POLYPHONY™ DIGITAL

# skysim

## 特徴

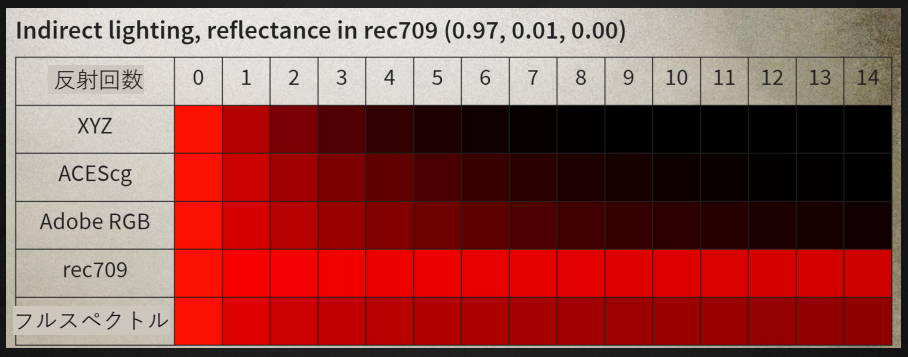
- フルスペクトルパストレーシング
- XYZ等色関数
- 太陽光スペクトル
- 地面色の設定
- Mie散乱位相関数
- 最適化

POLYPHONY™ DIGITAL

# skysim：フルスペクトルパストレーシング

空のレンダリングを行うにあたり、通常のRGBによる3chレンダリングでは不十分。よって、光の波長ごとの強度を計算する。

一般に、反射や散乱回数の多いレンダリングはフルスペクトルで行わないと誤差が大きくなる 。




https://jo.dreggn.org/path-tracing-in-production/2017/talk-jo/

# skysim：XYZ等色関数

散乱シミュレーションは波長単位のエネルギーについて行う。しかし最終的なレンダリングに使うにはRGBに変換する必要がある

- まずスペクトルからXYZ色空間に変換し、これをRGB（Rec.2020）色空間に変換するのが定石
- スペクトルからXYZ色空間に変換する際は波長方向の畳み込みを行うが、これに使うXYZ等色関数としてCIE (2006) 2-degを用いた[cmfs]

**スペクトル ▶ XYZ色空間 ▶ RGB色空間**

POLYPHONY DIGITAL

# skysim：XYZ等色関数

波長$\lambda$に対する放射輝度$L(\lambda)$が得られた時、等色関数$\bar{x}(\lambda)\ \bar{y}(\lambda)\ \bar{z}(\lambda)$を用いて以下の積分を計算するとXYZ色空間上の値が求まる。これをRGB色空間上の値に変換するのは単純な線形変換でよい

$$x = \int_{400}^{700} L(\lambda)\bar{x}(\lambda)d\lambda$$

$$y = \int_{400}^{700} L(\lambda)\bar{y}(\lambda)d\lambda$$

$$z = \int_{400}^{700} L(\lambda)\bar{z}(\lambda)d\lambda$$

$$\mathrm{R} = R_o x + R_1 y + R_2 z$$

$$\mathrm{G} = G_o x + G_1 y + G_2 z$$

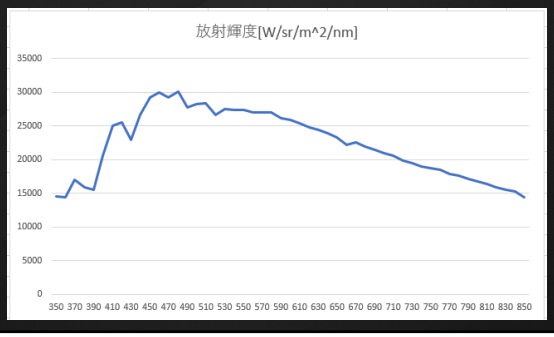
$$\mathrm{B} = B_o x + B_1 y + B_2 z$$

POLYPHONY DIGITAL

# skysim：太陽光スペクトル

太陽のスペクトルデータはASTM G173-03を採用

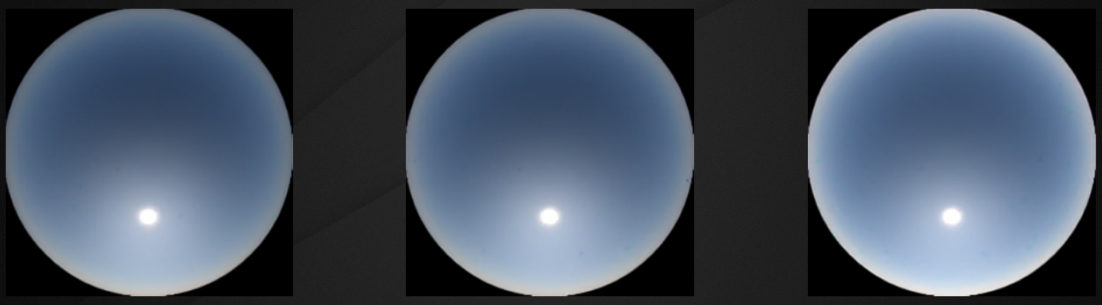
- Extraterrestrial Radiation(大気圏外)の値を使用。ただし、Irradianceとして提供されているので10[nm]単位で平均をとり太陽の立体角6.807e-5[sr]で割ることで平均放射輝度に変換している。
- 大気散乱計算時、太陽は平行光源として近似。今回の用途では十分であった。




POLYPHONY DIGITAL

## skysim：地面色の設定

空のレンダリング時、地面の色を設定できるようにした。これにより地面からの照り返しが反映される。意外と重要



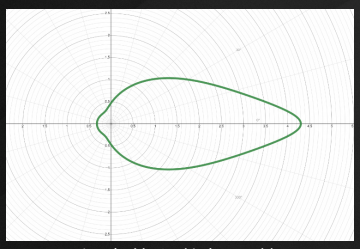
Albedo 0.0　Albedo 0.1　Albedo 0.5

POLYPHONY DIGITAL

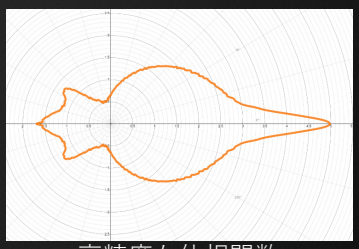
フルスペクトルなので地面色にもRGBが使えずスペクトルで定義する必要があった。面倒だったが、森林、砂漠、海などの代表的なスペクトルは見付けられた

# skysim：Mie散乱位相関数

- 既存手法は位相関数として低周波で単純なものを使う傾向があるということだったが、あまり印象が良くなかった。
- 最終的な印象を決定する要因としてMie散乱位相関数が重要であると考え、エアロゾル粒子分布から位相関数を高精度に計算して用いることで高周波な特徴が得られた。



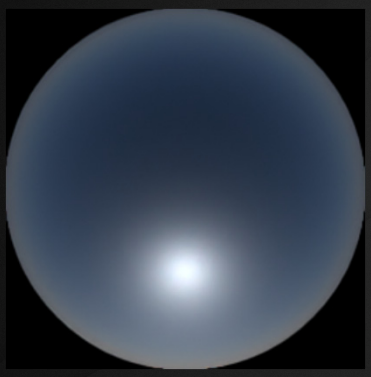
従来的な位相関数



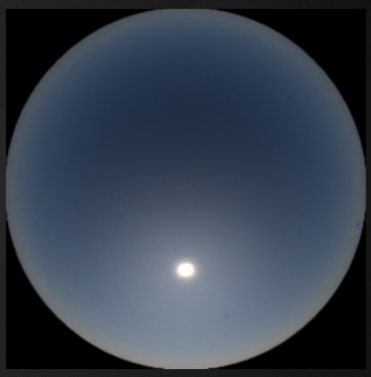
高精度な位相関数

POLYPHONY DIGITAL

# skysim：Mie散乱位相関数



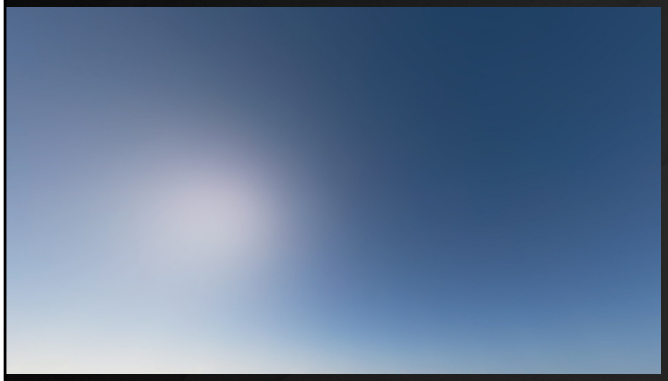
従来手法



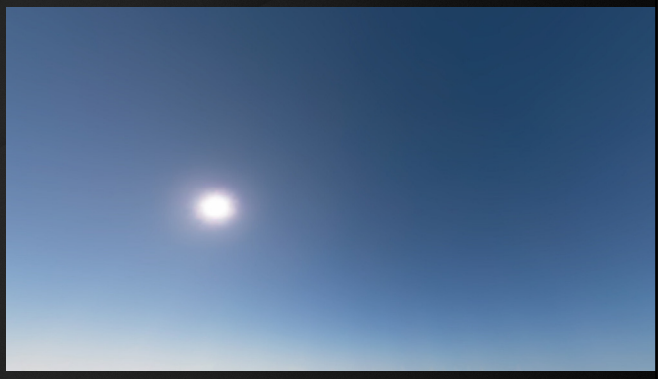
高精度

POLYPHONY DIGITAL

# skysim：Mie散乱位相関数

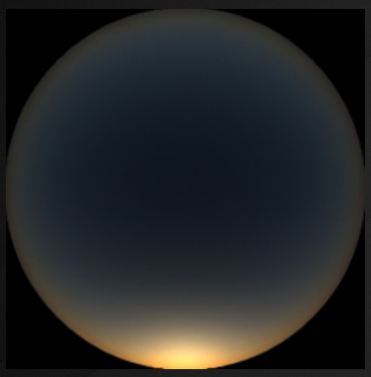


従来手法

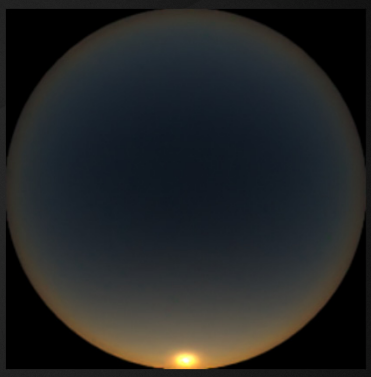


高精度

# skysim：Mie散乱位相関数



従来手法



高精度

POLYPHONY DIGITAL

# skysim：Mie散乱位相関数

従来手法

高精度

# skysim：最適化

パストレーシングによるボリュームレンダリング方程式の計算は一般に非常に計算コストが高いので、様々な最適化を行っている

# skysim：アルゴリズムの最適化

単純なパストレーシングをベースにしつつ、以下を実装

- Ratio-tracking による減衰率の推定 [Novák 2014]
- 位相関数の分布を用いた散乱方向のインポータンスサンプリング
- 分光視感効率を用いた波長のインポータンスサンプリング
- 最終結果に対するデノイザの適用
- ノイズの出やすい太陽近傍角度のサンプル増加（アダプティブサンプリング）
- Etc

POLYPHONY™ DIGITAL

# skysim：計算の最適化

アルゴリズム以外の部分に関して、以下のような最適化を試みた

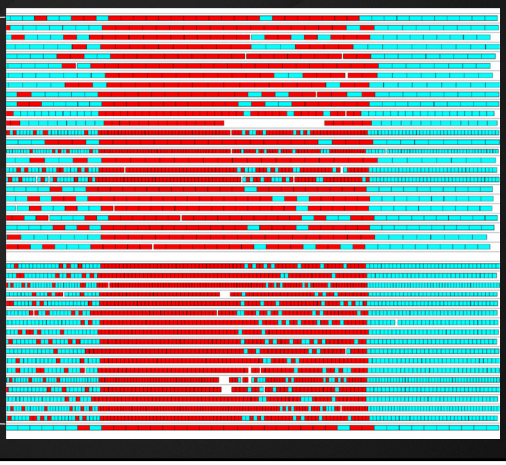
- プロファイルベースのコンパイル時最適化
  - 10%程度計算時間減少
- ピクセル単位でのマルチスレッド対応
- 複数のPCを利用した分散レンダリング

POLYPHONY™ DIGITAL

# skysim：計算の最適化

- 複数のPCを利用した、分散レンダリング
  - 開発用PCのうち、空いているPCを自由に使用できるインフラを整備
  - このインフラで得られた巨大な計算資源を活用した

1行=開発スタッフのPC・1台分



POLYPHONY DIGITAL

結果はこんな感じです

# 結果

印象的な夕陽が再現された

# 結果

印象的な夕陽が再現された

# 結果

バイモーダル分布による太陽近傍の複雑な散乱により、ボディに映り込む太陽も今までにない美しいものとなった

太陽近傍の散乱光は画面上ではエフェクトに埋もれやすく判りづらいですが、リフレクションでは太陽が少し大きくなったような効果があって、ボディの艶を色っぽく見せてくれます

# 本日の内容

1. 空と雲の物理
2. 現代的な大気散乱シミュレーションと関連研究
3. GT7における天球レンダリング手法(オフライン)
4. **GT7における天球レンダリング手法(ランタイム)**
5. まとめ

それでは、ここからはグラフィックスプログラマーの鈴木が、ランタイム側の天球レンダリング手法についてお話しします。

ランタイムというのは、つまりゲーム中のリアルタイムレンダリングのことになります。

# GT7の天球レンダリング手法
## （ランタイム）

※ランタイム：ゲーム中、リアルタイムに計算する処理

POLYPHONY DIGITAL

# 実装コンセプト

かなりの時間を費やしたが、大気モデリングと散乱シミュレーションとの精度を追求することで品質が大きく向上する手応えを得た

- この品質をランタイムでも実現したい
- しかしランタイムでこの計算量は到底まわらない

以上の条件から、skysimの計算結果を**Look-up Table**にする方針を考えた

POLYPHONY™ DIGITAL

最初にコンセプトの説明です。
ここまで研究開発にかなりの時間を費やしましたが、大気モデリングと散乱シミュレーションとの精度を追求することで空のレンダリング品質が大きく向上する手応えを得ました。

# 実装コンセプト

かなりの時間を費やしたが、大気モデリングと散乱シミュレーションとの精度を追求することで品質が大きく向上する手応えを得た

- この品質をランタイムでも実現したい
- しかしランタイムでこの計算量は到底まわらない

以上の条件から、skysimの計算結果を**Look-up Table**にする方針を考えた

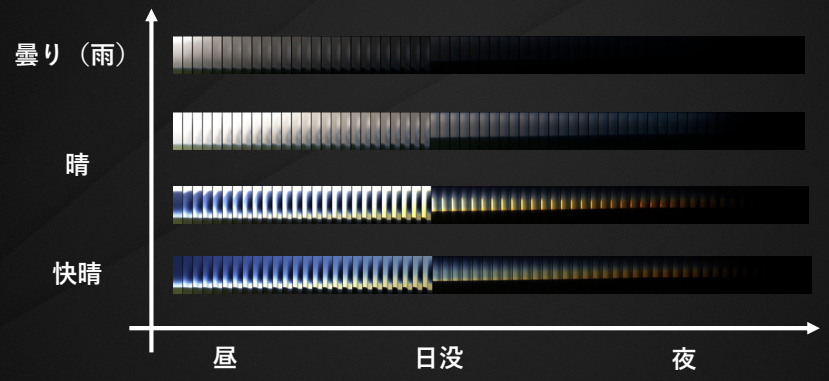
POLYPHONY™ DIGITAL

もちろんこの品質をランタイム、ゲーム中でも実現したいわけです。
しかしランタイムではこの計算量は到底まわりません。

そこで、skysimの計算結果をLook-up Tableにする方針を考えました。

# Look-up Table

- 最終的なゲームにおいて動的に変化するのは**時刻と天候**。よって、時刻と天候の二軸で空をレンダリングしてLUTに変換した。
  - GTの高度変化は300m程度なので空色を大きく変えるには至らない。よって高度変化はLUTからは除外できる。




POLYPHONY DIGITAL

グランツーリスモ 7 において、動的に変化するのは時刻と天候です。

よって、時刻と天候の二軸で空をレンダリングしてLook-up Tableに変換しました。
グランツーリスモの高度変化は300m程度なので、Look-up Tableのパラメータから除外できます。

## Look-up Table（時刻）

- 3D LUTテクスチャに全方向の空からの放射輝度が格納される。
  - 事前計算された高品質な空レンダリング結果を保持できる。
  - 単一の天候状態に対応。

- **時刻（太陽高度）方向 (1D)**と**全球方向 (2D)**についてアダプティブにマッピング、テクスチャ解像度（メモリ）を節約。
  - 時刻（太陽高度）方向：　64
  - 全球方向：　32 x 128
  - フォーマット：　RGB10,10,11 Float
    - 一枚当たり晴LUTで約1.0 MiB、曇LUTで約0.25 MiB

POLYPHONY DIGITAL

Look-up tableについてもう少し詳しく説明します。

我々のLook-up Tableは、3Dのテクスチャになっており、全方向の空からの放射輝度が格納されています。
これにより、事前計算された高品質な空レンダリング結果を保持できます。

一個のテクスチャは単一の天候状態に対応させます。

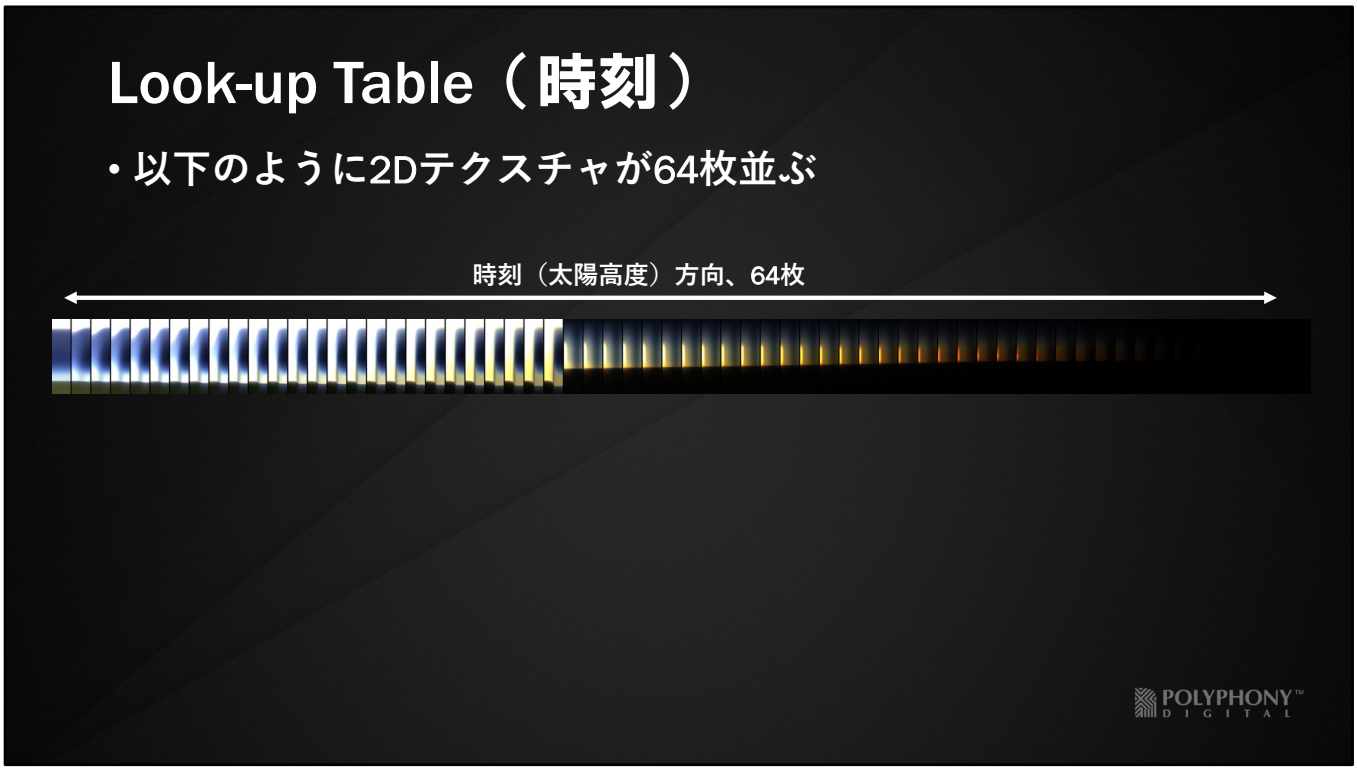
# Look-up Table（時刻）

- 3D LUTテクスチャに全方向の空からの放射輝度が格納される。
  - 事前計算された高品質な空レンダリング結果を保持できる。
  - 単一の天候状態に対応。

- **時刻（太陽高度）方向 (1D)**と**全球方向 (2D)**についてアダプティブにマッピング、テクスチャ解像度（メモリ）を節約。
  - 時刻（太陽高度）方向：　64
  - 全球方向：　32 x 128
  - フォーマット：　RGB10,10,11 Float
    - 一枚当たり晴LUTで約1.0 MiB、曇LUTで約0.25 MiB

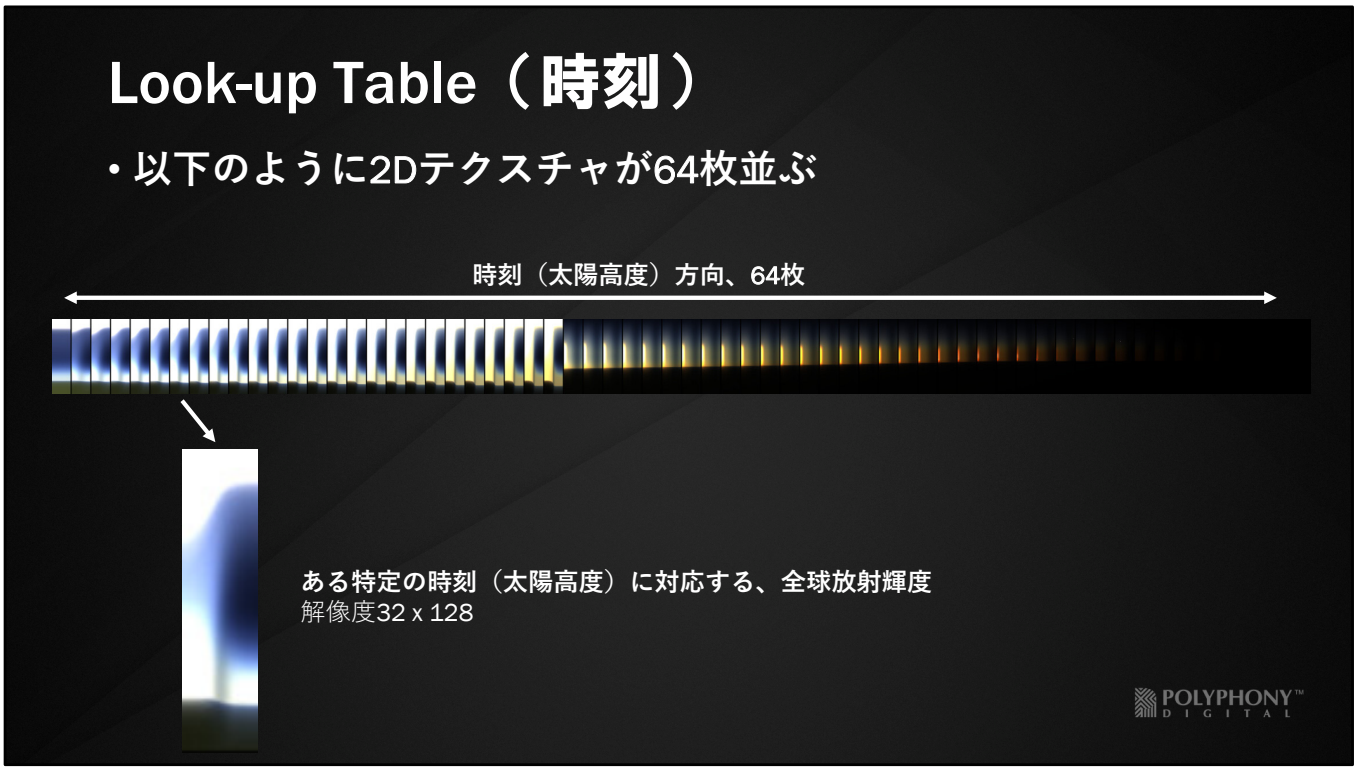
POLYPHONY DIGITAL

3Dテクスチャと言いましたが、時刻（太陽高度）方向の１次元と全球方向の2次元について、アダプティブにマッピングを行い、テクスチャ解像度を節約しています。




具体的なLook-up Tableは図のようになっています。

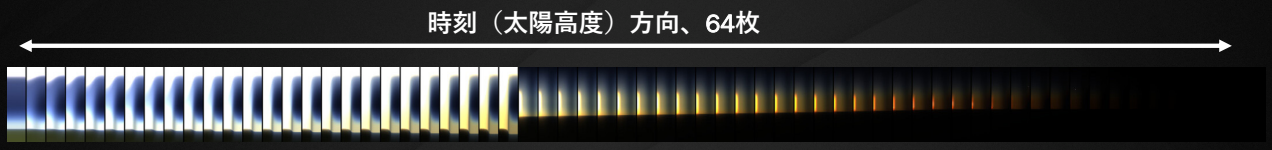
2次元のテクスチャが時刻方向に64枚並んでいます。

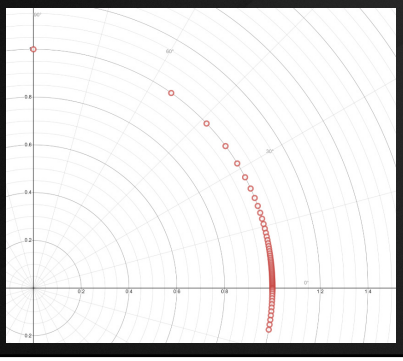



各スライスは、ある特定の時刻に対応する、全球放射輝度を格納しています。

# Look-up Table（時刻）

- **時刻（太陽高度）についてのアダプティブマッピング**
  - 水平線付近の高度に対応するテクスチャが多くなるようにする。



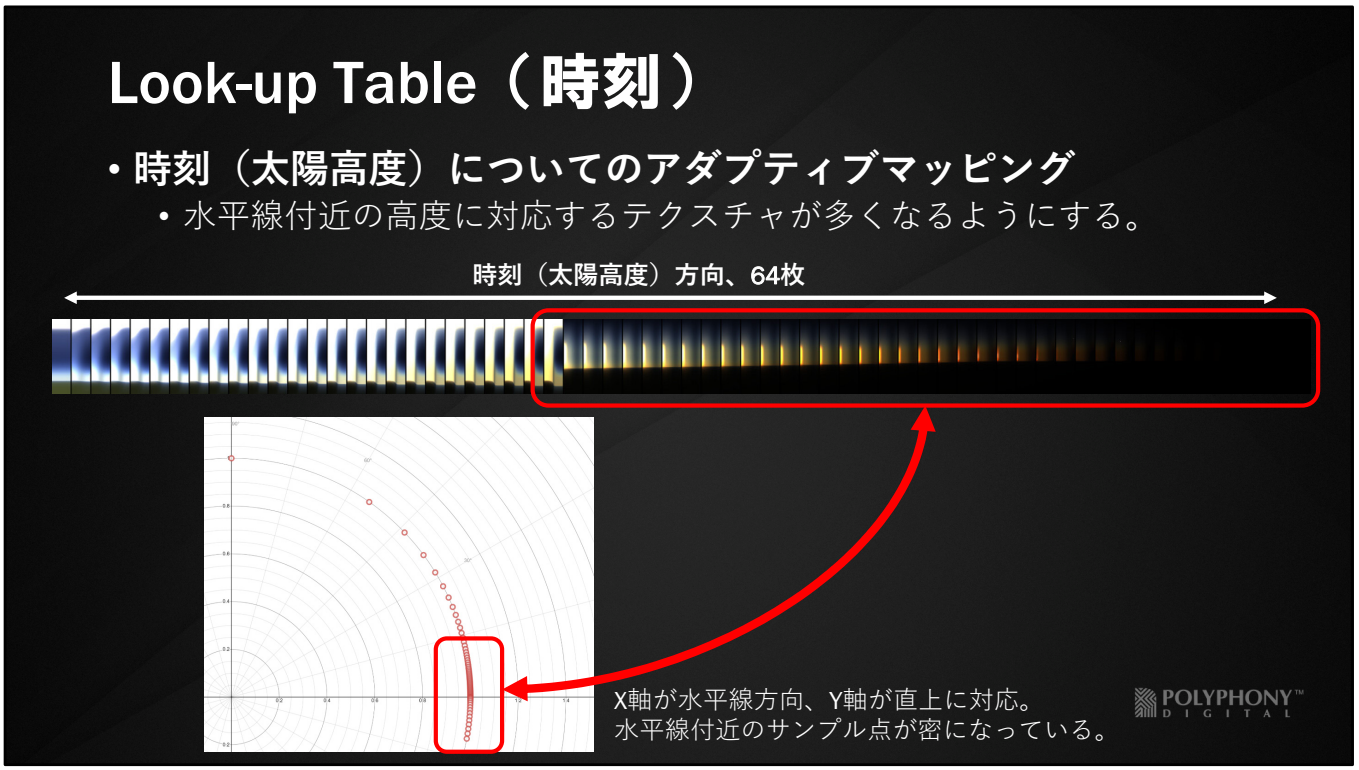



X軸が水平線方向、Y軸が直上に対応。
水平線付近のサンプル点が密になっている。

POLYPHONY DIGITAL

テクスチャを効率的に使うために、時刻つまり太陽高度に対してアダプティブにマッピングしています。

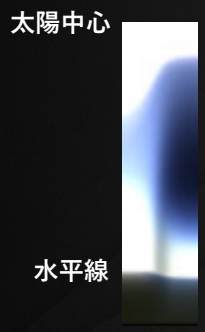
具体的には、水平線付近の高度に対して、より多くテクスチャが割り当てられるようになっています。




ご覧のように、水平線方向にテクスチャが密に割り当てられていることが分かると思います。

# Look-up Table（時刻）

- **全周方向についてのアダプティブマッピング**
  - 太陽方向を固定化したパラメタライズにすることで時刻方向のLUT枚数が少なくても滑らかに補間できるようにする。
  - 太陽周辺と水平線方向に多くのテクセルが割り当てられるようにする。

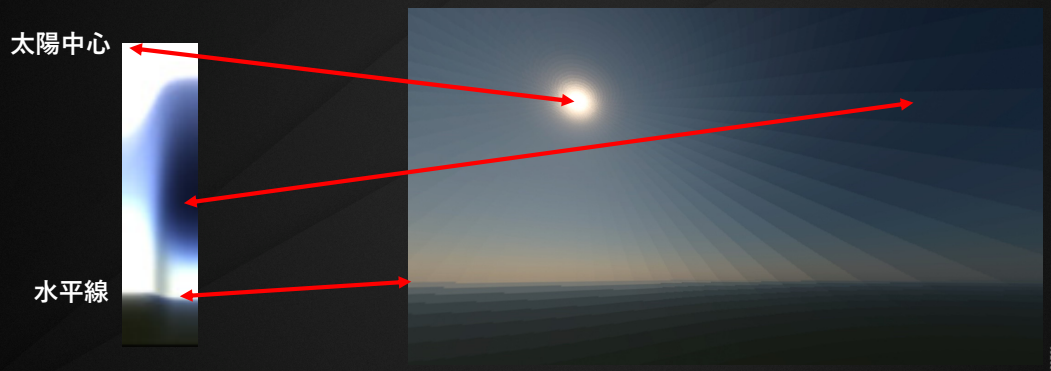



POLYPHONY DIGITAL

全周方向については、

# Look-up Table（時刻）

- **全周方向についてのアダプティブマッピング**
  - 太陽方向を固定化したパラメタライズにすることで時刻方向のLUT枚数が少なくても滑らかに補間できるようにする。
  - 太陽周辺と水平線方向に多くのテクセルが割り当てられるようにする。

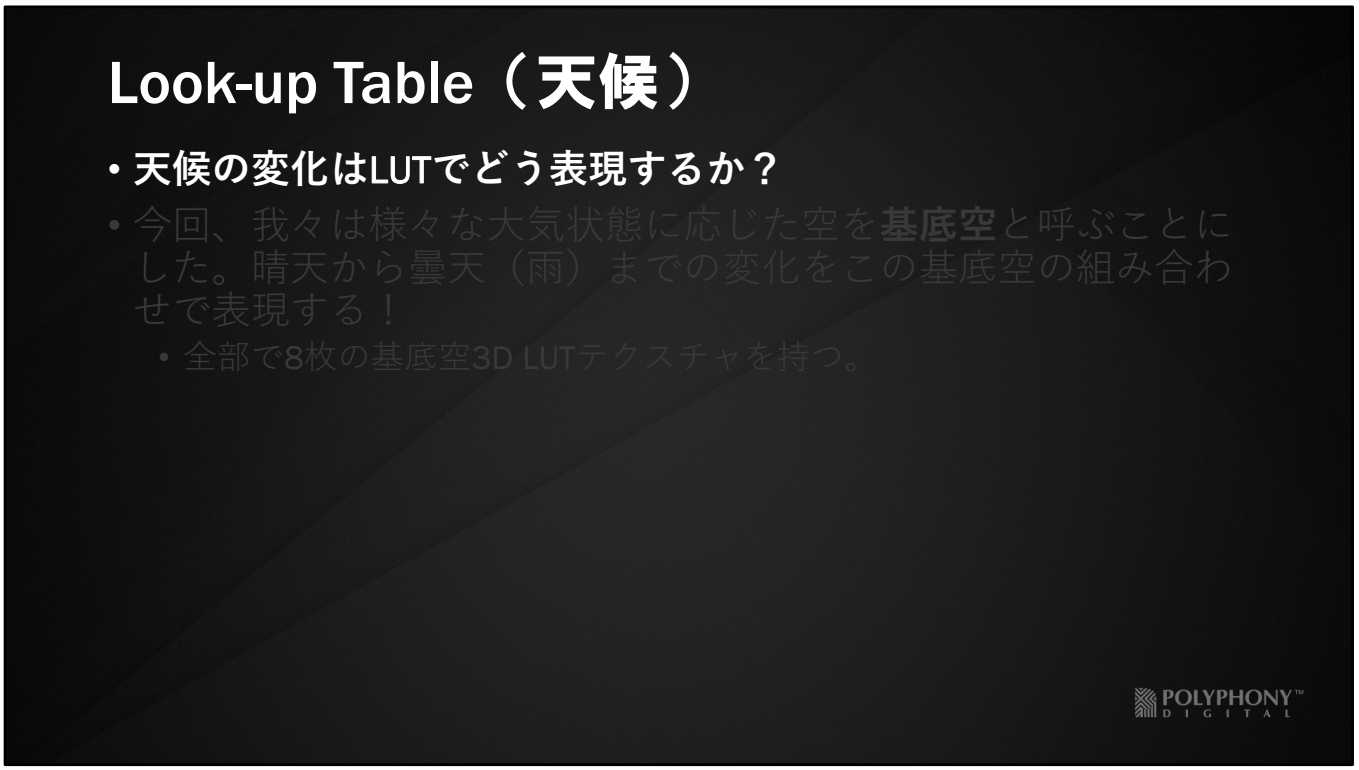



LUTのテクセルを可視化

POLYPHONY DIGITAL

LUTのテクセルを可視化してみました。

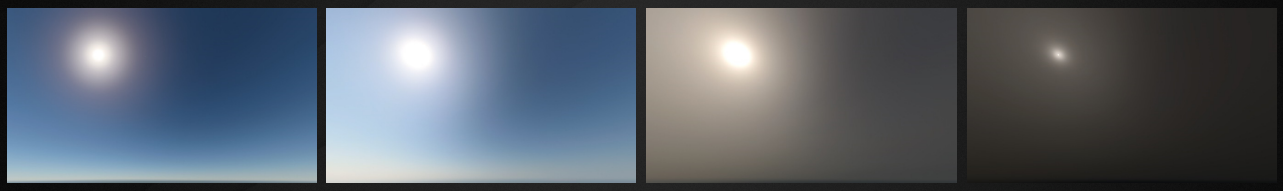
左側のLook-up Tableの各テクセルが、画面上では矢印のように対応してあります。
太陽周辺と水平線方向に、より多くのテクセルが割り当てられるようになっています。




さて、時刻と全周方向の空の放射輝度はLUTに格納されました。では、天候の変化はLUTでどう表現するか、というのが次の問題です。

## Look-up Table（天候）

- **天候の変化はLUTでどう表現するか？**
- 今回、我々は様々な大気状態に応じた空を**基底空**と呼ぶことにした。晴天から曇天（雨）までの変化をこの基底空の組み合わせで表現する！
  - 全部で8枚の基底空3D LUTテクスチャを持つ。



快晴　晴　曇天

POLYPHONY DIGITAL

今回、我々は様々な大気状態に応じた空を基底空と呼ぶことにしました。

晴天から曇天までの変化をこの基底空の組み合わせで表現します。全部で８枚の基底空LUTテクスチャを持ちます。

# 基底空Look-up Table

- 晴天のための基底空
  - エアロゾル分布は二個の極端な分布のみ与える。
    - エアロゾルの多様性をclearとhazedの二個の基底の線形補間で表現する。
    - 一軸方向の制御になる。

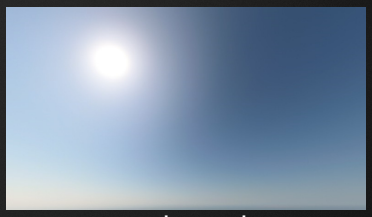
POLYPHONY DIGITAL

まず晴天のための基底空についてです。

# 基底空Look-up Table

- **晴天のための基底空**
  - エアロゾル分布は二個の極端な分布のみ与える。
    - エアロゾルの多様性をclearとhazedの二個の基底の線形補間で表現する。
    - 一軸方向の制御になる。

clear



hazed

POLYPHONY DIGITAL

晴天を表現する際のエアロゾル分布について、ここでは二個の極端な分布のみを与えて、LUTとしています。

エアロゾルの多様性をclearとhazedの二個の基底の線形補間で表現しているわけですね。

次に、曇天のための基底空についてです。

空を一様に灰色にするような高空の巨大な雲を曇天用の基底空として表現しています。

つまり、大気モデルとして巨大な雲を与えています。これは、エアロゾル分布などを工夫することで実現しています。

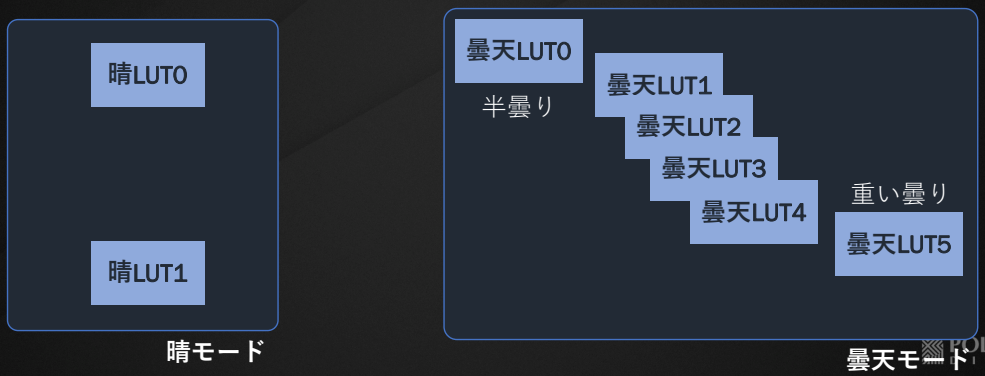
# 基底空Look-up Table

- **曇天（雨）のための基底空**
  - 空を一様に灰色にするような高空の巨大な雲（高層雲）を曇天用の基底空として表現。
  - 高層雲の発生過程では散乱が急激に変化することから、細かく基底を用意（6個）することで補間精度が向上、自然な遷移となった。

このような高層雲の発生過程では散乱が急激に変化することから、６個の基底を用意することで補間精度が向上、空の表情の自然な遷移を表現できました。

# 基底空Look-up Tableの補間

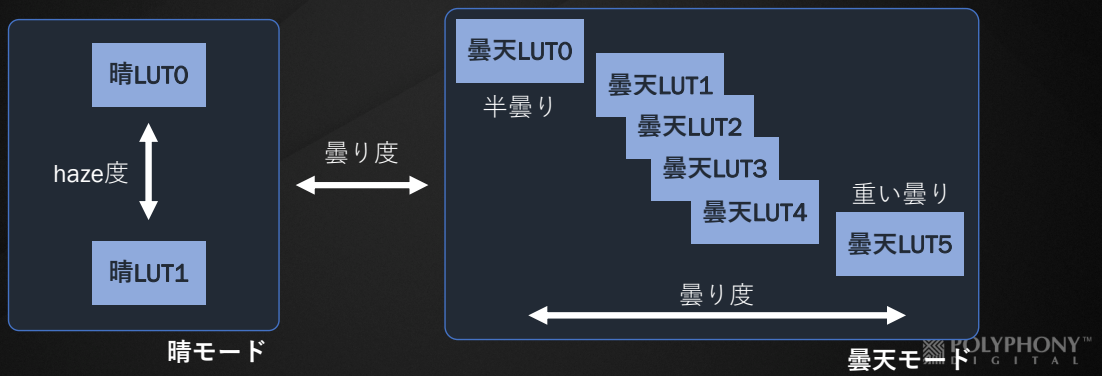
- **晴天用の基底（２個）**と**曇天用の基底（６個）**を得たら、それらのレンダリング結果をLUTに変換する。
- 各LUTを天候パラメータ（haze度、曇り度）に基づいて補間して、最終的なレンダリングに用いる。




以上のように、晴天用の基底２個と曇天用の基底６個を得たら、それらのレンダリング結果をLUTに変換しておきます。

# 基底空Look-up Tableの補間

- **晴天用の基底（２個）**と**曇天用の基底（６個）**を得たら、それらのレンダリング結果をLUTに変換する。
- 各LUTを天候パラメータ（haze度、曇り度）に基づいて補間して、最終的なレンダリングに用いる。

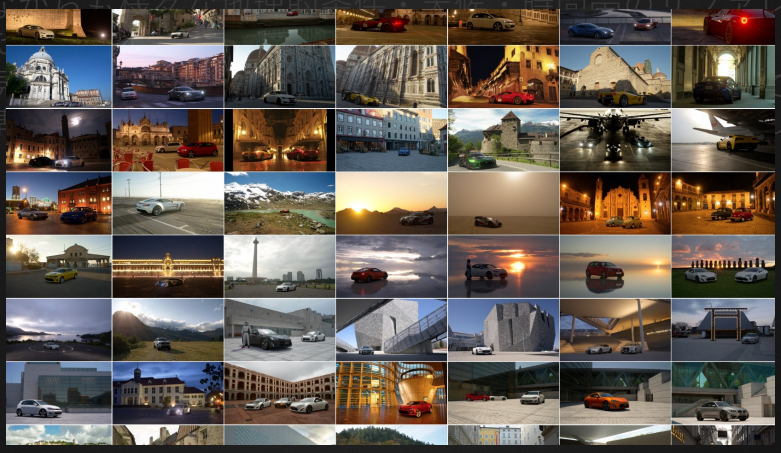



各LUTを、別途実装されている気象シミュレーションからもたらされる天候パラメータであるhaze度・曇り度に基づいて補間して、最終的なレンダリングに用います。

補間のためのウェイトはテクニカルアーティストによって調節されています。

## 2. 基底空Look-up Table

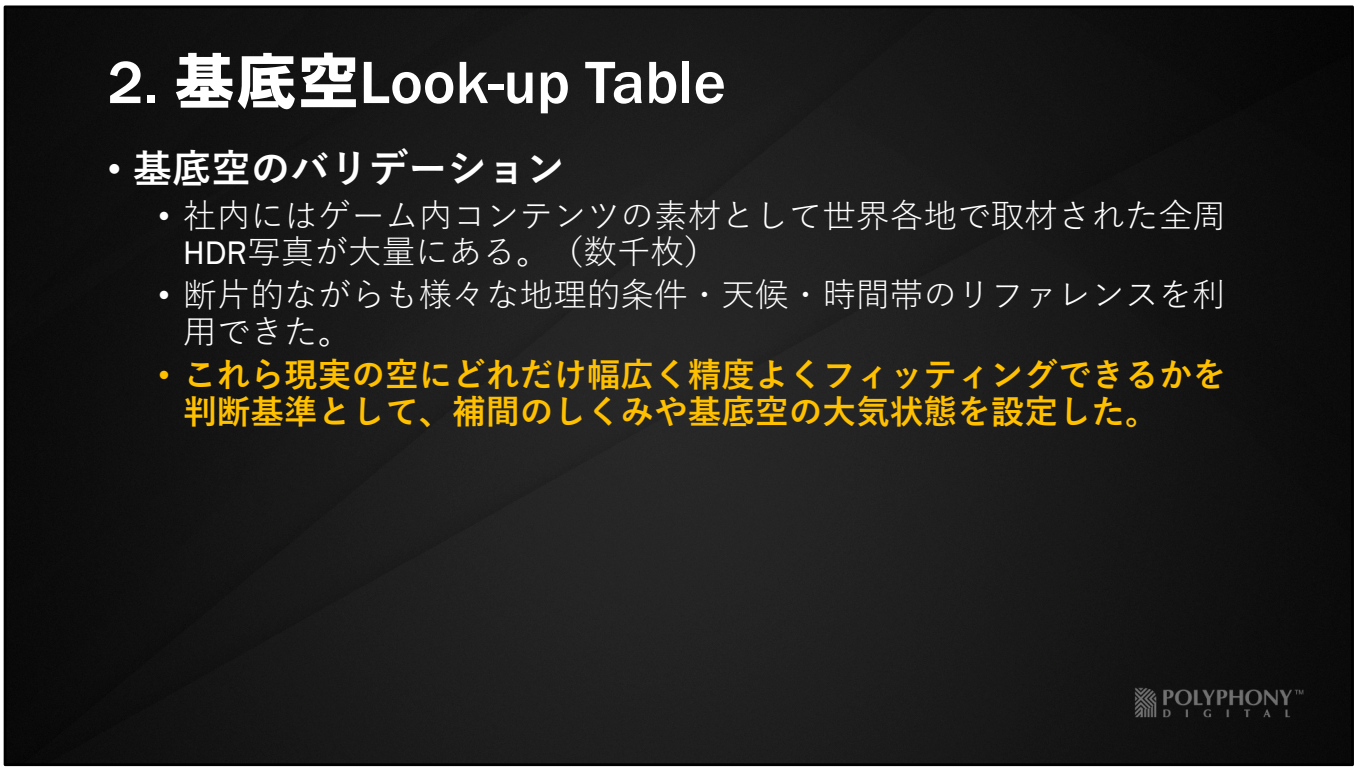
- **基底空のバリデーション**
  - 社内にはゲーム内コンテンツの素材として世界各地で取材された全周HDR写真が大量にある。（数千枚）
  - 断片的ながらも様々な地理的条件、天候・時間帯のリファレンスを利用できた
  - これら現
    判断基準



POLYPHONY DIGITAL

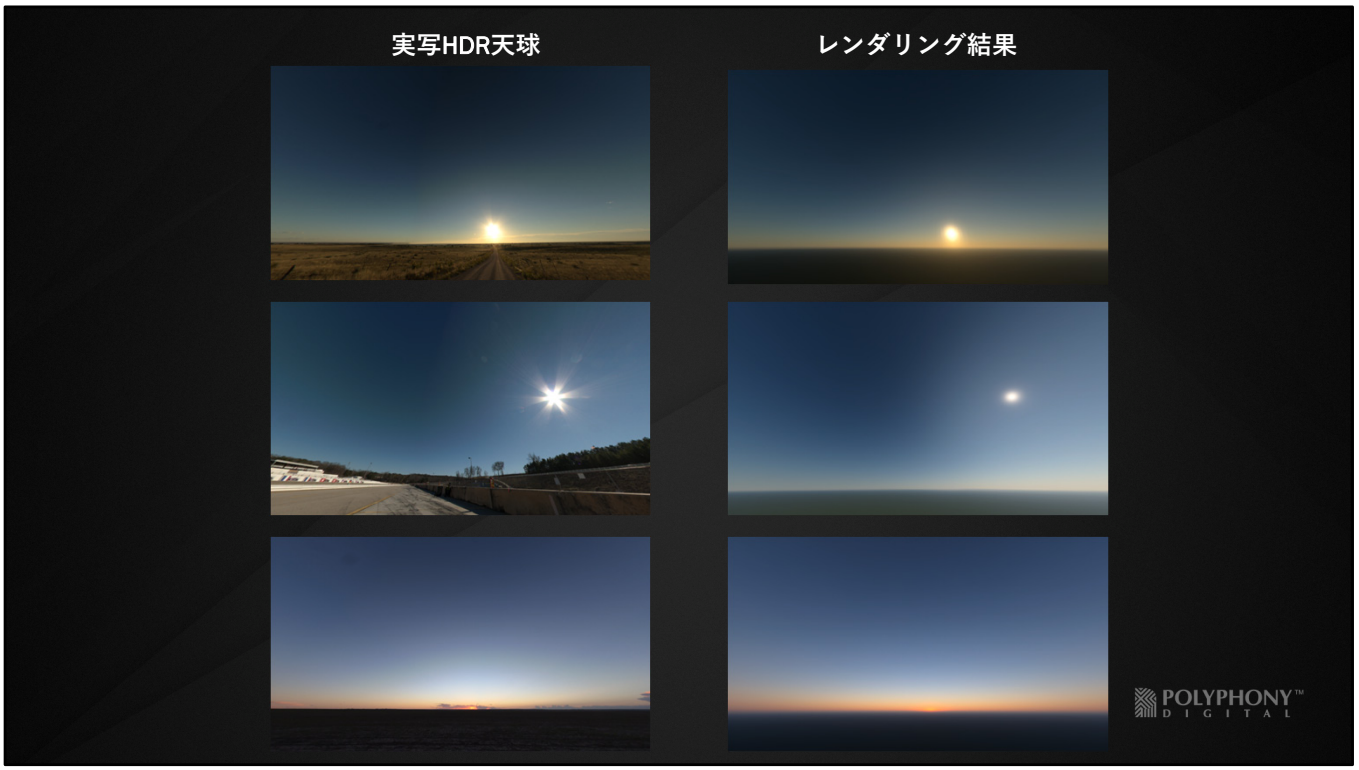
このように設定された基底空は、実際の空の写真を用いてバリデーションされました。

社内にはゲーム内コンテンツの素材として世界各地で取材された全周HDR写真が大量にあります。




こういった素材により、断片的ながらも様々な地理的条件・天候・時間帯のリファレンスを利用できました。

これら現実の空にどれだけ幅広く、精度よくフィッティングできるかを判断基準として、テクニカルアーティストの手によってここまで説明してきたような、補間のしくみや基底空の大気状態が設定されました。




これは実写とレンダリング結果の比較です。左が実写、右がレンダリング結果ですね。

ご覧のように、幅広い実写画像を、今回のシステムでうまく再現できていることが分かると思います。

話は前後しますが、このように雲がある場合や、曇天のバリデーションも同様に行いました。

やはり、うまく再現できていると思います。

# その他のLook-up Table

- 空のLUT以外にも以下を計算して各種処理に用いた
  - フォグのLUT（二種類）
  - 大気による光減衰関数のLUT
  - 太陽光（並行光源）の強度のLUT
  - 空のアンビエントのLUT
- LUTを利用することでランタイムの計算コストを減らすことができた。
  - これはPS4などでは特に重要。

POLYPHONY DIGITAL

さて、ここまで空のLook-up Tableを紹介してきました。

実際には空以外にもフォグ、減衰関数、太陽光、アンビエントなどのLook-up Tableを用いています。
これらのLook-up Tableを利用することでランタイムの計算コストを減らすことができました。

ここまで空のレンダリングの説明をしてきましたが、フォグについても簡単に触れておこうと思います。

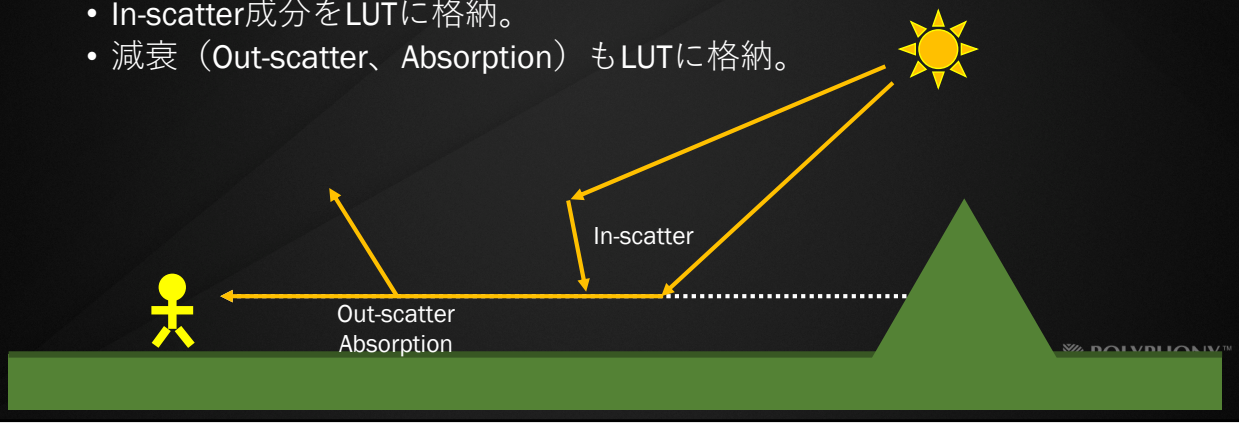
# フォグ

- フォグはすでに説明した通り大気散乱として計算可能。ただし空の場合と違って手前で打ち切られる。
- 無限遠においては、フォグは空と同じ輝度になる。よって、フォグの計算は空の大気モデルと一貫していてほしい。
  - In-scatter成分をLUTに格納。
  - 減衰（Out-scatter、Absorption）もLUTに格納。

フォグは大気散乱として計算可能です。ただし空の場合と違って手前で打ち切られます。

# フォグ

- フォグはすでに説明した通り大気散乱として計算可能。ただし空の場合と違って手前で打ち切られる。
- 無限遠においては、フォグは空と同じ輝度になる。よって、フォグの計算は空の大気モデルと一貫していてほしい。
  - In-scatter成分をLUTに格納。
  - 減衰（Out-scatter、Absorption）もLUTに格納。




無限遠においては、フォグは空と同じ輝度になります。よって、フォグの計算は空の大気モデルと一貫する必要があります。

そこで。すでに説明したように、各成分はLook-up Tableに格納します。
具体的にはIn-scatter成分と、減衰成分を格納します。

# フォグ

- 最終的に二つのフォグ用LUTを用いて中遠景のInscatter成分を計算。
  - 二つのLUTは、「**雲による遮蔽を考慮したフォグ**」と「**考慮してないフォグ**」のそれぞれを事前計算した結果となる。
  - フォグの色と遠近感は大幅に改善し、リファレンスにも近付いた。
- これにより、雲の切れ間に印象的なライトシャフトが再現されることにもなった。

POLYPHONY DIGITAL

最終的に二つのフォグ用LUTを用いて中遠景のInscatter成分を計算しました。
この二つのLook-up Tableは、それぞれ雲による遮蔽を考慮したフォグと考慮してないフォグの結果となります。

# フォグ

- 最終的に二つのフォグ用LUTを用いて中遠景のInscatter成分を計算。
  - 二つのLUTは、「**雲による遮蔽を考慮したフォグ**」と「**考慮してないフォグ**」のそれぞれを事前計算した結果となる。
  - フォグの色と遠近感は大幅に改善し、リファレンスにも近付いた。
- これにより、雲の切れ間に印象的なライトシャフトが再現されることにもなった。

これはゲームのキャプチャです。

ご覧のように、雲の間から差し込む光の筋が分かると思います。これは、説明したフォグによって得られている効果です。

これは他の例です。

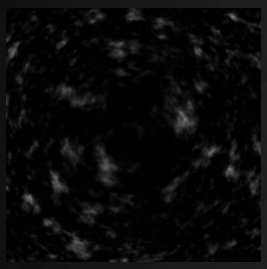
次に雲のレンダリングについてです。

# 雲のモデリング

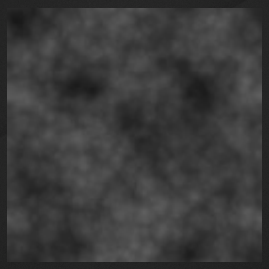
まずは雲のモデリングについてです。

# 雲モデリング

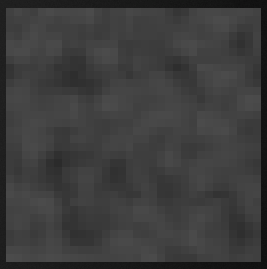
- 今回、多様な「雲」のうち積雲のみレイマーチングで表現。
- 積雲は以下の3つのノイズを後述の Coverage Texture で合成してモデリングされている。



積雲マップ(2D)



低周波ノイズ(3D)



高周波ノイズ(3D)

POLYPHONY DIGITAL

今回、多様な「雲」のうち積雲のみレイマーチングで表現します。

積雲は図のような、３つのノイズを後述の Coverage Texture で合成してモデリングされています。

それぞれの効果を見ていきましょう。

まず、積雲マップとCoverageTexだけ与えた場合の絵です。

雲に見えないこともないですが、すこしぼんやりしています。

これに、低周波ノイズを加えます。

さらに、スケールを変調した中周波ノイズを足します。

雲らしくなってきたのではないでしょうか。

最後に高周波ノイズを足して、ディテールを調節しています。

もう一度、ノイズを加える前の雲の絵を出してみます。

# 雲モデリング

- Coverage Texture
  - 以下の4chをRGBAにパックしている（最終解像度は 256x128）
  - 横軸が積雲マップ由来の密度、縦軸が高度に対応し、ノイズ(3D)と組み合わせて最終的な三次元密度場を生成する。

Height成分　低周波成分　中周波成分　アンビエント成分

POLYPHONY DIGITAL

ノイズによるモデリングは、CoverageTextureとよばれる別のテクスチャを用いて制御しています。

図のような4chがテクスチャにパックされており、積雲マップ由来の密度と高度を組み合わせてテクスチャを参照します。

参照結果と先述のノイズが組み合わさることで、最終的な三次元密度場を生成します。

次に雲のライティングについてです。

# 雲ライティング

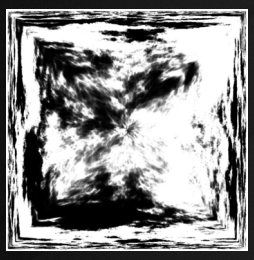
- 既存手法と同様のやり方でレイマーチングを行い雲の散乱計算。
- 以下のような雲G-bufferを計算（上半球）
  - レイマーチング自体は負荷分散のため複数フレームにまたがって計算されるため毎フレーム更新できない一方、ライティング自体は毎フレーム更新したい。よって要素分離する。
  - 各チャンネルにライティングに必要な情報を格納しておく。

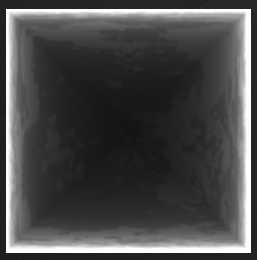
POLYPHONY™ DIGITAL

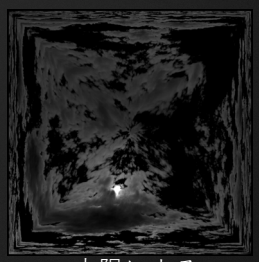
ライティングは、既存手法と同様のやり方でレイマーチングを行い、雲の内部の散乱を計算します。

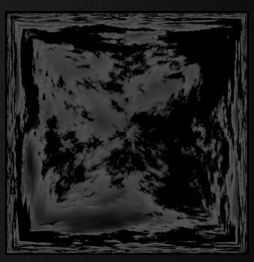
# 雲ライティング

- 既存手法と同様のやり方でレイマーチングを行い雲の散乱計算。
- 以下のような**雲G-buffer**を計算（上半球）
  - レイマーチング自体は負荷分散のため複数フレームにまたがって計算されるため毎フレーム更新できない一方、ライティング自体は毎フレーム更新したい。よって要素分離する。
  - 各チャンネルにライティングに必要な情報を格納しておく。


背景（空）に対する透過率


雲までの距離


太陽によるライティング（in-scatter）


アンビエントライティング

このとき、直接絵を出力せず、図のような雲G-bufferを出力します。

というのも、レイマーチング自体は負荷分散のため複数フレームにまたがって計算されるため毎フレーム更新できない一方、ライティング自体は毎フレーム更新したいからです。

この需要を満たすために、最終的な絵ではなく、ライティング要素を分離されたG-bufferを出力するようにしています。

# 雲ライティング

- **雲G-bufferを用いてライティングを行う。**
  - 太陽光によるSingle-scatteringを愚直に計算。
  - Multiple-scatteringは[Wrenninge et al. 2013]によるアドホック近似を採用。

Single-scatteringのみ

Multiple-scattering追加

最終的な絵を出力する時は、雲G-bufferを用いて、太陽光によるSingle-Scatteringを愚直に計算します。

ただし、Wrenningeらによる手法を用いてMultiple-Scatteringをアドホックに近似します。

# 本日の内容

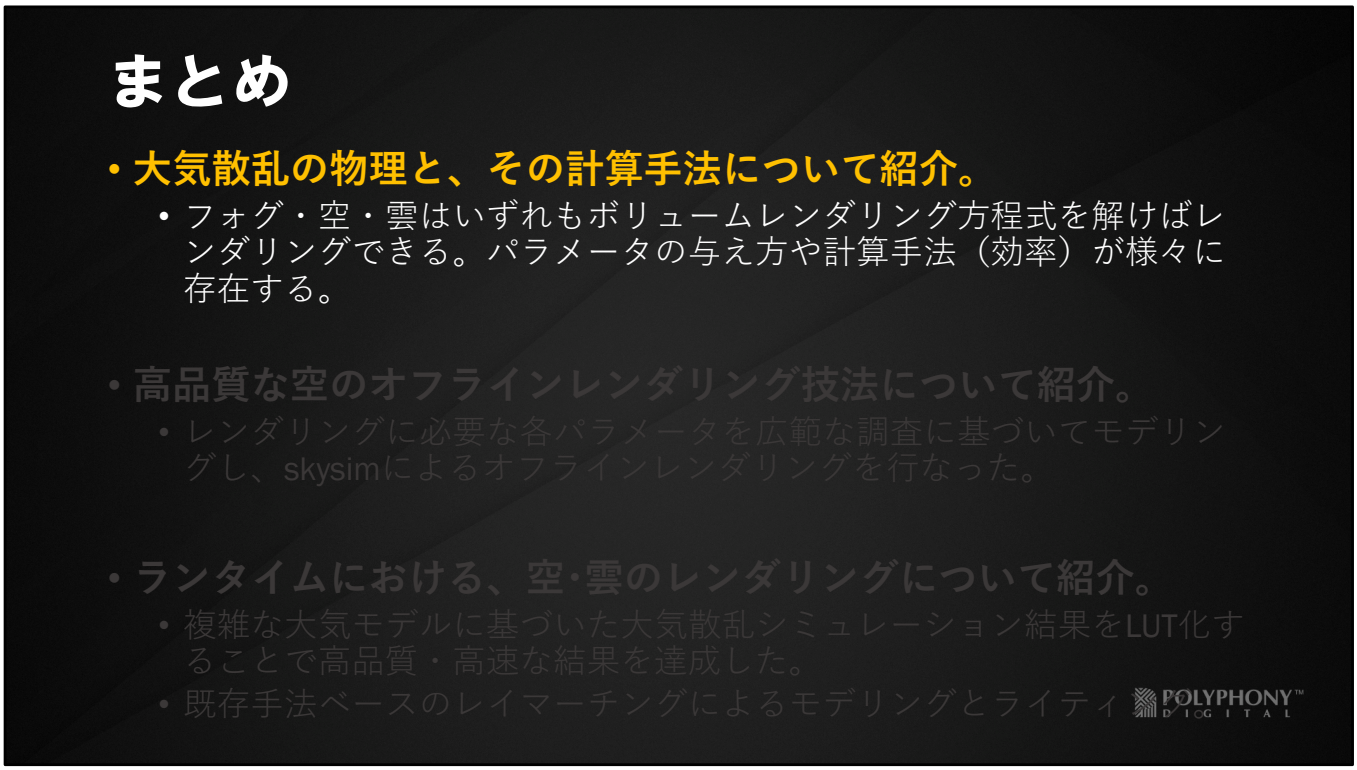
1. 空と雲の物理
2. 現代的な大気散乱シミュレーションと関連研究
3. GT7における天球レンダリング手法(オフライン)
4. GT7における天球レンダリング手法(ランタイム)
5. **まとめ**

POLYPHONY DIGITAL

それでは最後に、本日の発表のまとめに移りたいと思います。

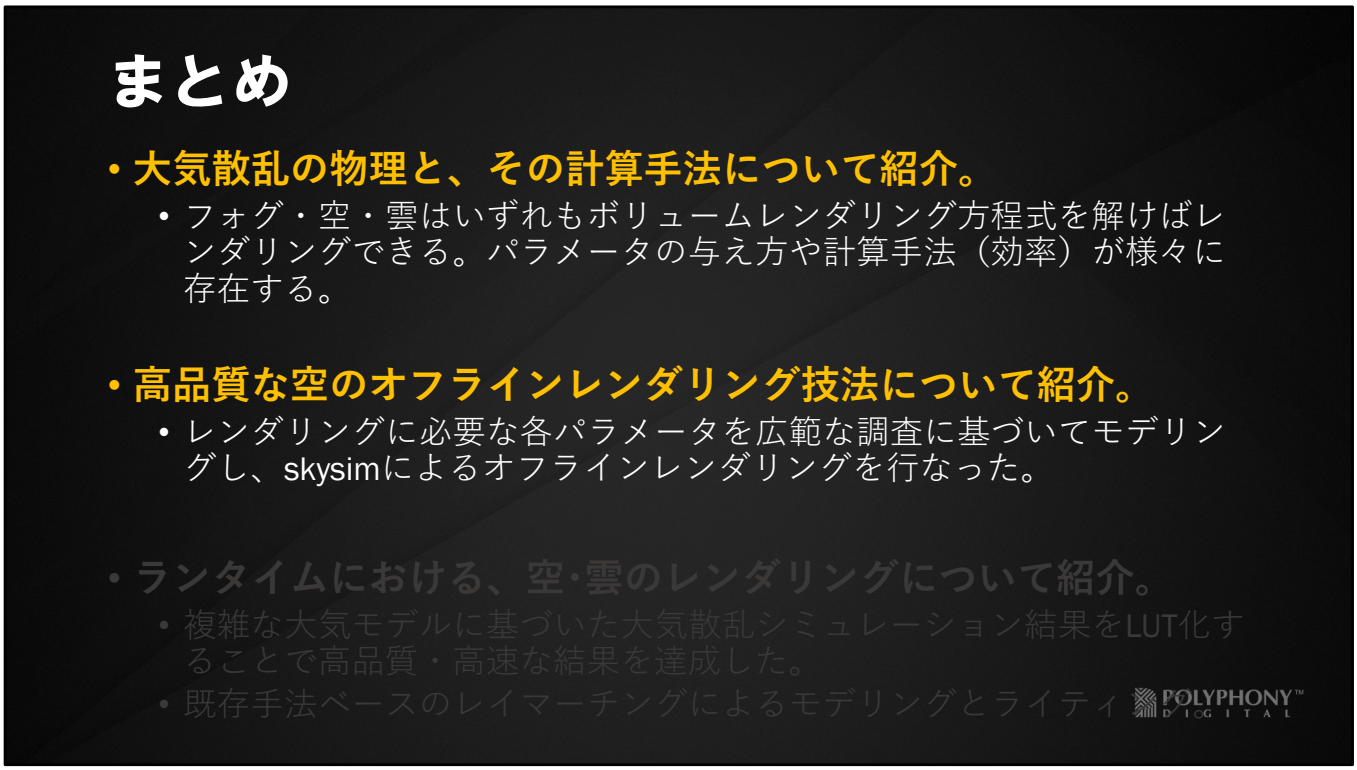
# まとめ

POLYPHONY™ DIGITAL




最初に、大気散乱の物理とその計算手法について紹介しました。

フォグ・空・雲はいずれもボリュームレンダリング方程式を解けばレンダリングできます。
そして、そのパラメータの与え方や計算手法（効率）は様々に存在しています。




次に、高品質な空のオフラインレンダリング技法について紹介しました。

このようなレンダリングに必要な各種のパラメータは、我々独自の広範な調査に基づいてモデリングされました。
そして、skysimと呼ばれるソフトウェアを開発し、オフラインでレンダリングを行いました。

# まとめ

- **大気散乱の物理と、その計算手法について紹介。**
  - フォグ・空・雲はいずれもボリュームレンダリング方程式を解けばレンダリングできる。パラメータの与え方や計算手法（効率）が様々に存在する。
- **高品質な空のオフラインレンダリング技法について紹介。**
  - レンダリングに必要な各パラメータを広範な調査に基づいてモデリングし、skysimによるオフラインレンダリングを行なった。
- **ランタイムにおける、空･雲のレンダリングについて紹介。**
  - 複雑な大気モデルに基づいた大気散乱シミュレーション結果をLUT化することで高品質・高速な結果を達成した。
  - 既存手法ベースのレイマーチングによるモデリングとライティング。

POLYPHONY DIGITAL

最後に、ランタイム、つまりゲーム中におけるLook-up Tableによる空のレンダリングおよび雲のレンダリングついて紹介しました；

複雑な大気モデルに基づいた大気散乱シミュレーション結果をLUT化することで、高品質かつ高速な空のレンダリングを達成しました。
また、雲のレンダリングは、既存手法ベースのレイマーチングによるモデリングとライティングを行いました。

# 参考文献


- [Novák et al. 2018] Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. 2018. Monte Carlo Methods for Volumetric Light Transport Simulation. Computer Graphics Forum (Proceedings of Eurographics - State of the Art Reports) 37, 2 (May 2018).
- [Nishita et al. 1993] Tomoyuki Nishita, Takao Sirai, Katsumi Tadamura, and Eihachiro Nakamae. 1993. Display of the Earth Taking into Account Atmospheric Scattering. In Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93), Association for Computing Machinery, Anaheim, CA, 175–182. DOI:https://doi.org/10.1145/166117.166140
- [Nishita et al. 1996] Tomoyuki Nishita, Yoshinori Dobashi, and Eihachiro Nakamae. 1996. Display of Clouds Taking into Account Multiple Anisotropic Scattering and Sky Light. In Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96), Association for Computing Machinery, New York, NY, USA, 379–386. DOI:https://doi.org/10.1145/237170.237277
- [Bruneton and Neyret 2008] Eric Bruneton and Fabrice Neyret. 2008. Precomputed Atmospheric Scattering. Computer Graphics Forum (2008). DOI:https://doi.org/10.1111/j.1467-8659.2008.01245.x
- [Hosek and Wilkie 2012] Lukas Hosek and Alexander Wilkie. 2012. An Analytic Model for Full Spectral Sky-Dome Radiance. ACM Trans. Graph. 31, 4 (July 2012). DOI:https://doi.org/10.1145/2185520.2185591
- [Hillaire 2020] Sébastien Hillaire. 2020. A Scalable and Production Ready Sky and Atmosphere Rendering Technique. Computer Graphics Forum 39, 4 (2020), 13–22. DOI:https://doi.org/https://doi.org/10.1111/cgf.14050
- [Wilkie et al. 2021] Alexander Wilkie, Petr Vevoda, Thomas Bashford-Rogers, Lukáš Hošek, Tomáš Iser, Monika Kolářová, Tobias Rittig, and Jaroslav Křivánek. 2021. A Fitted Radiance and Attenuation Model for Realistic Atmospheres. ACM Trans. Graph. 40, 4 (July 2021). DOI:https://doi.org/10.1145/3450626.3459758


POLYPHONY™ DIGITAL

参考文献を付けておきました。興味のある方はご参照ください。

発表は以上になります。ご静聴、ありがとうございました。

# 参考文献


- [Novák 2014] Jan Novák, Andrew Selle, and Wojciech Jarosz. 2014. Residual Ratio Tracking for Estimating Attenuation in Participating Media. ACM Trans. Graph. 33, 6 (November 2014). DOI:https://doi.org/10.1145/2661229.2661292
- [cmfs] Colour & Vision Research Laboratory. http://cvrl.ucl.ac.uk/cmfs.htm
- [Krueger and Minzner 1976] A. Krueger and Raymond Minzner. 1976. A MidLatitude Ozone Model for the 1976 U.S. Standard Atmosphere. Journal of Geophysical Research 81, (August 1976), 4477–4481. DOI:https://doi.org/10.1029/JC081i024p04477
- [Omar et al. 2005] Ali H. Omar, Jae-Gwang Won, David M. Winker, Soon-Chang Yoon, Oleg Dubovik, and M. Patrick McCormick. 2005. Development of global aerosol models using cluster analysis of Aerosol Robotic Network (AERONET) measurements. Journal of Geophysical Research: Atmospheres 110, D10 (2005). DOI:https://doi.org/https://doi.org/10.1029/2004JD004874
- [O3 Spectra] https://www.iup.uni-bremen.de/gruppen/molspec/databases/referencespectra/o3spectra2011/index.html
- [Bohren and Huffman, 1983] Craig Bohren and Donald Huffman. 1998. Absorption and Scattering of Light by Small Particles. Wiley Science Paperback Series.
- [Schneider 2015] Andrew Schneider. 2015. The Real-time Volumetric Cloudscapes of Horizon: Zero Dawn. Advances in Real Time Rendering, Part I. In ACM SIGGRAPH 2015 Courses (SIGGRAPH '15), Association for Computing Machinery, Los Angeles, California. DOI:https://doi.org/10.1145/2776880.2787701


POLYPHONY DIGITAL

# 参考文献


- [Schneider 2017] Andrew Schneider. 2017. Nubis: Authoring Real-Time Volumetric Cloudscapes with the Decima Engine. Advances in Real-Time Rendering, Part I. In ACM SIGGRAPH 2017 Courses (SIGGRAPH '17), Association for Computing Machinery, Los Angeles, California. DOI:https://doi.org/10.1145/3084873.3096476
- [Hillaire 2016] Sébastien Hillaire. 2016. Physically Based Sky, Atmosphere & Cloud Rendering in Frostbite. Physically Based Shading in Theory and Practice. In ACM SIGGRAPH 2016 Courses (SIGGRAPH '16), Association for Computing Machinery, Anaheim, California. DOI:https://doi.org/10.1145/2897826.2927353
- [Bauer 2019] Fabian Bauer. 2019. Creating the Atmospheric World of Red Dead Redemption 2: A Complete and Integrated Solution. Advances in Real-Time Rendering in Games, Part 2. In ACM SIGGRAPH 2019 Courses (SIGGRAPH '19), Association for Computing Machinery, Los Angeles, California. DOI:https://doi.org/10.1145/3305366.3335036
- [Wrenninge et al. 2013] Magnus Wrenninge, Chris Kulla, and Viktor Lundqvist. 2013. Oz: The Great and Volumetric. In ACM SIGGRAPH 2013 Talks (SIGGRAPH '13), Association for Computing Machinery, Anaheim, California. DOI:https://doi.org/10.1145/2504459.2504518


POLYPHONY™ DIGITAL